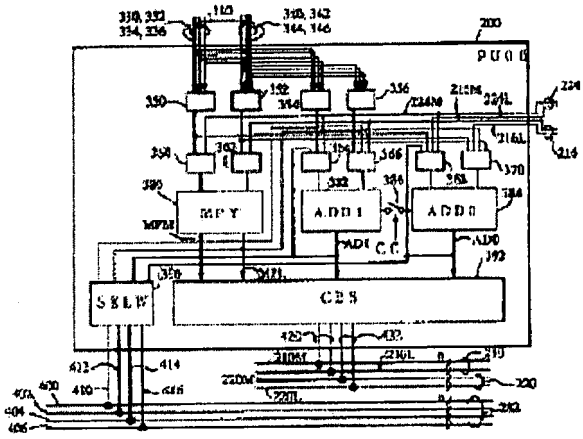
AX5080 Reference 1

PARALLEL PROCESSING PROCESSOR, ITS PROCESSING UNIT AND OPERATING METHOD FOR THE PARALLEL PROCESSING PROCESSOR

Publication number:	JP7064789
Publication date:	1995-03-10
Inventor:	NAKASE YASUNOBU
Applicant:	MITSUBISHI ELECTRIC CORP
Classification:	
- international:	***G06F9/38;*** **G06F9/38;** (IPC1-7): G06F9/38
- european:	
Application number:	JP19930210783 19930825
Priority number(s):	JP19930210783 19930825

Abstract of **JP7064789**

PURPOSE:To provide a general purpose parallel processing processor which can easily be controlled. CONSTITUTION:Each of the plural processing units of the processor includes computing elements 380, 382 and 384, selectors 360 to 370 for giving the high/low order bits of input data buses 216 and 224 from another unit to these computing elements and a cross bar switch 392 for outputting the outputs of the computing elements 380, 382 and 384 to optional one of data buses 210 and 220 to another unit. The connection of respective selectors is previously set so as to execute various arithmetic operation. On the other hand, data switching between with another processing unit by way of the data bus realizes various arithmetic operation as the whole processor. The constitution of each processing unit is the same, easily laid out and can be controlled by interchangeble control instruction.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開平7－64789
(43)公開日　平成７年(1995)３月10日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　9/38	３７０　Ａ			

審査請求　未請求　請求項の数19　ＯＬ　（全 29 頁）

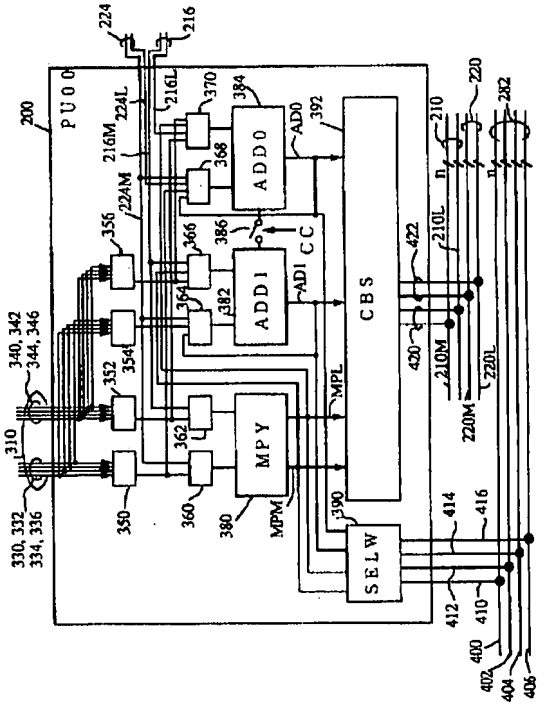
(21)出願番号　特願平5－210783

(22)出願日　平成５年(1993)８月25日

(71)出願人　000006013
三菱電機株式会社
東京都千代田区丸の内二丁目２番３号
(72)発明者　中瀬　泰伸
兵庫県伊丹市瑞原４丁目１番地　三菱電機株式会社エル・エス・アイ研究所内
(74)代理人　弁理士　深見　久郎　（外３名）

(54)【発明の名称】　並列処理プロセッサおよびそのプロセッシングユニットならびにこの並列処理プロセッサの動作方法

(57)【要約】
【目的】　簡単に制御可能な、汎用的並列処理プロセッサを提供する。
【構成】　プロセッサの複数個のプロセッシングユニットの各々は、演算器３８０、３８２、３８４と、他のユニットからの入力データバス２１６、２２４の上位ｎビットまたは下位ｎビットをこれら演算器に与えるためのセレクタ３６０～３７０と、演算器３８０、３８２、３８４の出力を他のユニットへのデータバス２１０、２２０の任意のものに出力するためのクロスバースイッチ３９２とを含む。各セレクタの接続を予め設定することにより様々な演算を実行できる。また他のプロセッシングユニットとのデータバスを介したデータ交換によりプロセッサ全体として多様な演算を実現できる。各プロセッシングユニットの構造は同一でレイアウト容易であり、互換性のある制御命令で制御できる。



1

【特許請求の範囲】

【請求項1】　それぞれ複数個の入力を有し、与えられるデータの間に所定の演算を行なって結果を出力するための複数個の演算手段と、

複数の単方向入力データバスに接続され、前記複数個の演算手段の各入力ごとに、前記複数の単方向入力データバスのうちのいずれか1つを可制御的に選択して、選択された単方向入力データバスを介して与えられるデータの一部を前記入力に与えるための入力データバス選択手段と、

前記複数個の演算手段の出力に接続された入力と、前記単方向入力データバスと同じ数の単方向出力データバスに接続された出力とを有し、前記演算手段の出力の各々を、前記単方向出力データバスのいずれかに出力するための出力データバス選択手段と、

前記複数個の演算手段により所望の複合演算を実現するために、前記入力データバス選択手段と、前記出力データバス選択手段とによるデータの経路を制御するための制御手段とを含む、並列処理のためのプロセッシングユニット。

【請求項2】　前記複数個の演算手段が、

2つのnビット幅の入力を有し、与えられる2つのデータを乗算して2nビットの結果を出力する乗算器と、

各々が2つのnビット幅の入力を有し、与えられる2つのデータを加算してnビット幅の結果を出力する2つの加算器とを含む、請求項1に記載の並列処理のためのプロセッシングユニット。

【請求項3】　前記複数の単方向入力データバスおよび前記複数の単方向出力データバスの各々は2nビット幅を有し、

前記入力データバス選択手段は、前記複数個の演算手段の各入力ごとに前記複数の単方向入力データバスのうちのいずれか1つを可制御的に選択して、選択された単方向入力データバスを介して与えられるデータの上位または下位のnビットを前記入力に与えるための手段を含み、

前記出力データバス選択手段は、前記演算手段の出力の各々を、前記単方向出力データバスの任意のいずれかの上位nビットまたは下位nビットまたはその双方に出力可能とするための手段を含む、請求項2に記載の並列処理のためのプロセッシングユニット。

【請求項4】　前記2つの加算器の一方はキャリー出力を有し、

前記2つの加算器の他方はキャリー入力を有し、

さらに、前記キャリー出力と前記キャリー入力とを可制御的に断続するための手段を含む、請求項2に記載の並列処理のためのプロセッシングユニット。

【請求項5】　前記入力データバス選択手段は、前記2つの加算器の1つの少なくとも1つの入力について、前記複数の単方向入力データバスのうちのいずれか1つまたは該加算器自身の出力のいずれかを可制御的に選択して、選択された単方向入力データバスを介して与えられるデータまたは該加算器自身の出力のいずれかの一部を前記入力に与えるための手段を含む、請求項2に記載の並列処理のためのプロセッシングユニット。

2

【請求項6】　前記入力データバス選択手段が、前記2つの加算器の1つの少なくとも1つの入力について、前記複数の単方向入力データバスのうちのいずれか1つまたは前記乗算器の出力のいずれかを可制御的に選択して、選択された単方向入力データバスを介して与えられるデータまたは前記乗算器の出力のいずれかの一部を前記入力に与えるための手段を含む、請求項2に記載の並列処理のためのプロセッシングユニット。

【請求項7】　前記入力データバス選択手段が、前記2つの加算器の1つの少なくとも1つの入力について、前記複数の単方向入力データバスのうちのいずれか1つまたは該加算器自身の出力または前記乗算器の出力の一部のいずれかを可制御的に選択して、選択された単方向入力データバスを介して与えられるデータの一部または該加算器自身の出力または前記乗算器の出力の一部のいずれかを前記入力に与えるための手段を含む、請求項2に記載の並列処理のためのプロセッシングユニット。

【請求項8】　所定の情報を予め記憶するための読出専用記憶手段をさらに含み、

前記入力データバス選択手段が、前記乗算器の少なくとも1つの入力について、前記複数の単方向入力データバスのうちのいずれか1つまたは前記読出専用記憶手段の出力のいずれかを可制御的に選択して、選択された単方向入力データバスを介して与えられるデータまたは前記記憶手段の出力のいずれかの一部を前記入力に与えるための手段を含む、請求項2に記載の並列処理のためのプロセッシングユニット。

【請求項9】　n個のプロセッシングユニットと、

隣り合うプロセッシングユニットを所定方向に円環状に順次に接続するための第1の単方向データバスと、

1つおいて隣り合うプロセッシングユニットを円環状に順次に双方向に接続するための、第2の単方向データバスとを含み、

前記nは4のベキ乗であり、

各前記プロセッシングユニットは、

それぞれ複数個の入力を有し、与えられるデータの間に所定の演算を行なって結果を出力するための複数個の演算手段と、

隣接するプロセッシングユニットからの入力となる前記第1の単方向データバスと、前記第2の単方向データバスのうちの該プロセッシングユニットへの入力データバスとに接続され、前記複数個の演算手段の各入力ごとに前記第1および第2の単方向データバスのうちのいずれか1つを可制御的に選択して、選択された単方向データバスを介して与えられるデータの一部を前記入力に与え

るための入力データバス選択手段と、
前記複数個の演算手段の出力に接続された入力と、隣接するプロセッシングユニットへの出力となる前記第１の単方向データバスと、前記第２の単方向データバスのうちの該プロセッシングユニットからの出力データバスとに接続された出力とを有し、前記演算手段の出力の各々を、前記第１および第２の単方向データバスのいずれかに出力するための出力データバス選択手段と、
前記複数個の演算手段により所望の複合演算を実現するために、前記入力データバス選択手段と、前記出力データバス選択手段とによるデータの経路と、前記演算手段による演算の実行とを制御するための制御手段とを含む、並列処理プロセッサ。

【請求項１０】　前記プロセッシングユニットと同数の、各々が一度に２つのデータを出力可能なデータ記憶手段と、
前記データ記憶手段の各々と、前記プロセッシングユニットの各々とを接続するための複数の読出データバスおよび複数の書込データバスとをさらに含み、
各前記プロセッシングユニットにおいて、
前記入力データバス選択手段は、隣接するプロセッシングユニットからの入力となる前記第１の単方向データバスと、前記第２の単方向データバスのうちの該プロセッシングユニットへの入力データバスと、前記複数の読出データバスとに接続され、前記複数個の演算手段の各入力ごとに前記第１および第２の単方向データバスと前記読出データバスとのうちのいずれか１つを可制御的に選択して、選択されたデータバスを介して与えられるデータの一部を前記入力に与えるための手段を含み、
前記出力データバス選択手段は、前記複数個の演算手段の出力に接続された入力と、隣接するプロセッシングユニットへの出力となる前記第１の単方向データバスと、前記第２の単方向データバスのうちの該プロセッシングユニットからの出力データバスと、前記書込データバスとに接続された出力とを有し、前記演算手段の出力の各々を、前記データバスのいずれかに出力するための手段とを含む、請求項９に記載の並列処理プロセッサ。

【請求項１１】　前記プロセッシングユニットと同数の、各々が一度に２つのデータを出力可能なデータ記憶手段と、
前記データ記憶手段の各々と、前記プロセッシングユニットの各々とを接続するための複数の読出データバスおよび複数の書込データバスとをさらに含み、
前記複数のプロセッシングユニットは、各々が４の冪乗個のプロセッシングユニットを含む複数個のグループに分割されており、
前記複数のプロセッシングユニットと前記複数のデータ記憶手段とは１対１に対応付けられており、
各前記プロセッシングユニットにおいて、
前記入力データバス選択手段は、隣接するプロセッシングユニットからの入力となる前記第１の単方向データバスと、前記第２の単方向データバスのうちの該プロセッシングユニットへの入力データバスと、前記複数の読出データバスのうち該プロセッシングユニットが含まれるグループのプロセッシングユニットと対応付けられたデータ記憶手段からの読出データバスとに接続され、前記複数個の演算手段の各入力ごとに前記第１および第２の単方向データバスと前記読出データバスとのうちのいずれか１つを可制御的に選択して、選択されたデータバスを介して与えられるデータの一部を前記入力に与えるための手段を含み、
前記出力データバス選択手段は、前記複数個の演算手段の出力に接続された入力と、隣接するプロセッシングユニットへの出力となる前記第１の単方向データバスと、前記第２の単方向データバスのうちの該プロセッシングユニットからの出力データバスと、前記書込データバスのすべてとに接続された出力とを有し、前記演算手段の出力の各々を、前記データバスのいずれかに出力するための手段を含む、請求項９に記載の並列処理プロセッサ。

【請求項１２】　４個のプロセッシングユニットと、
隣り合うプロセッシングユニットを所定方向に円環状に順次に接続するための４本の第１の単方向データバスと、
１つおいて隣り合うプロセッシングユニットを双方向に接続するための、４本の第２の単方向データバスとを含み、
各前記プロセッシングユニットは、
各々ｎビットの２つの入力を有し、与えられるデータの間に乗算を行なって２ｎビット幅の結果を出力するための乗算手段と、
各々が、各々ｎビットの２つの入力を有し、与えられるデータの間に加算を行なってｎビット幅の結果を出力するための第１および第２の加算手段と、
前記第１の加算手段のキャリー出力を前記第２の加算手段のキャリー入力に可制御的に与えるためのキャリー切換手段と、
隣接するプロセッシングユニットからの入力となる前記第１の単方向データバスと、前記第２の単方向データバスのうちの該プロセッシングユニットへの入力データバスとに接続され、前記複数個の演算手段の各入力ごとに前記第１および第２の単方向データバスのうちのいずれか１つを可制御的に選択して、選択された単方向データバスを介して与えられるデータの一部を前記乗算手段と前記加算手段との前記入力に与えるための入力データバス選択手段と、
前記乗算手段の出力と前記加算手段の出力とに接続された入力と、隣接するプロセッシングユニットへの出力となる前記第１の単方向データバスと、前記第２の単方向データバスのうちの該プロセッシングユニットからの出

力データバスとに接続された出力とを有し、前記乗算手段および前記加算手段の出力の各々を、前記第1および第2の単方向データバスのいずれかに出力するための出力データバス選択手段と、

前記乗算手段および前記加算手段による所望の複合演算を実現するために、前記入力データバス選択手段と、前記出力データバス選択手段とによるデータの経路を制御するための制御手段と、

を含む並列処理プロセッサにおいて、所望の演算を行なうための動作方法であって、

演算に必要なデータの各々を前記入力データバス選択手段に与えるステップと、

前記入力データバス選択手段により、前記データの各々を上位および下位のnビットずつに分解し、前記4つのプロセッシングユニットの、前記乗算手段および前記加算手段の入力のいずれか2つにそれぞれ与えるステップと、

すべての前記プロセッシングユニットの前記キャリー切換手段を、前記所望の演算に応じて設定するステップと、

各前記プロセッシングユニットの前記出力データバス選択手段と前記入力データバス選択手段とを制御して、前記所望の演算が得られるように各前記プロセッシングユニットの前記乗算手段と、前記第1および第2の加算手段との間の接続を設定するステップと、

演算結果が、前記データバスのうちの所望のものに出力されるように、前記所望の演算によって定まる所定のプロセッシングユニットの前記乗算手段および加算手段のうちの所定のものの出力の、所定の部分を前記所望のデータバスに出力するように前記出力データバス選択手段を制御するステップとを含む、動作方法。

【請求項13】　前記接続を設定するステップは、

あるプロセッシングユニット内の乗算手段または加算手段の出力の上位nビットが、他のプロセッシングユニット内の乗算手段または加算手段の下位nビットに入力されるように、これらプロセッシングユニットの前記出力データバス選択手段および前記入力データバス選択手段によるデータ経路を設定するステップを含む、請求項12に記載の動作方法。

【請求項14】　前記接続を設定するステップは、

あるプロセッシングユニット内の乗算手段または加算手段の出力の上位nビットが、他のプロセッシングユニットへの出力データバスの下位nビットに出力されるように、該プロセッシングユニットの前記出力データバス選択手段によるデータ経路を設定するステップを含む、請求項12に記載の動作方法。

【請求項15】　各プロセッシングユニットごとに準備され、前記制御手段が実行する制御命令を格納するための複数の命令記憶手段をさらに含む、請求項9に記載の並列処理プロセッサ。

【請求項16】　前記制御命令は、対応のデータ記憶手段を制御するための第1の種類の制御命令と、前記制御手段による前記演算手段の制御のための第2の種類の制御命令との2つの系統に分類される、請求項15に記載の並列処理プロセッサ。

【請求項17】　前記第1の種類の制御命令は、前記データ記憶手段の各々の2つの読出アドレスと、1つの書込アドレスとを指定する、請求項16に記載の並列処理プロセッサ。

【請求項18】　前記制御手段は、与えられる第2の種類の制御命令が変更されるまでは、直前に与えられた第2の種類の制御命令に従って前記演算手段と前記入力データバス選択手段と前記出力データバス選択手段とを制御する、請求項16に記載の並列処理プロセッサ。

【請求項19】　前記命令記憶手段は、

複数個の命令を格納する命令メモリと、

命令メモリの読出アドレスを指定するためのプログラムカウンタと、

前記プログラムカウンタにより指定されたアドレスを先頭として2つの命令を一度に読出すための手段と、

読出された2つの命令が同一の系統に属するか否かを判断するための手段と、

判断結果に従って、前記制御手段または前記データ記憶手段またはその双方に制御命令を与えるための手段と、

判断結果に従って、前記プログラムカウンタのカウントを1または2増加させるための手段とを含む、請求項16に記載の並列処理プロセッサ。

【発明の詳細な説明】

【0001】

【産業上の利用分野】この発明は、複数のプロセッシングユニットにより構成される並列処理プロセッサの改良に関し、特に、幅広い演算に対応できる、制御の容易な並列処理プロセッサと、そのためのプロセッシングユニットと、並列処理プロセッサの動作方法とに関する。

【0002】

【従来の技術】従来の並列処理プロセッサ（以下単に「プロセッサ」と呼ぶ）の構成を図16および図17に示す。このプロセッサは、「ＩＳＳＣＣダイジェスト・オブ・テクニカル・ペーパーズ」（“ISSCC Digest of Technical Papers”、１９９１年２月、２５２～２５３頁）に発表されたプロセッサのブロック構成のうち、本願発明に関連する主要演算部分を抜粋し、簡略化して示したものである。

【0003】図16を参照して、このプロセッサは、4個のプロセッシングユニットＰＵ００～ＰＵ１１（図16中では符号３０、３２、３４、３６により示される）と、アドレス演算ユニット（ＡＵ）４８と、ワーキングメモリ３８と、データキャッシュメモリ４０、４２、４４、４６とを含む。各プロセッシングユニットからデータキャッシュメモリ４０、４２、４４、４６およびワー

キングメモリ３８へのアクセスは、４本のキャッシュメモリ読出専用バス５０と、６本の読出書込兼用バス４２との、合計１０本のバスを介して行なわれる。このプロセッサはさらに、レジスタファイル５８と、セレクタ５６とを含む。セレクタ５６とレジスタファイル５８とは、プロセッシングユニット間のデータ交換を行なうためのものであり、キャッシュメモリ読出専用バス５０と、読出書込兼用バス５２と、ＳＢＵＳ５４とから読込んだデータをレジスタファイル５８を介して各プロセッシングユニット３０、３２、３４、３６とバス５２、５４とに出力可能である。

【０００４】図１７を参照して、各プロセッシングユニット３０、３２、３４、３６は、類似ではあるが相互にやや異なった構成となっている。各プロセッシングユニット３０、３２、３４、３６は、演算器ＡＬＵ７０、７２、７４、７６と、乗算器ＭＰＹ８０、８２、８４、８６と、加算器ＡＤＤ９０、９２、９４、９６とを含む。また各プロセッシングユニット３０、３２、３４、３６内には、ＡＬＵ、乗算器、加算器への入力を選択するためのセレクタが含まれている。これについては後述するが、図１７においては、図面の簡略化のために、セレクタの入出力については簡略化して示してある。

【０００５】図１７を参照して、たとえばプロセッシングユニット３０は、５：１のセレクタ１１０と１１２とを含む。プロセッシングユニット３０はさらに、ＡＬＵ７０の出力の一方と、レジスタファイル５８からの出力と、セレクタ１１０の出力とから１つを選択してＡＬＵ７０の一方の入力に与えるためのセレクタ１３０と、セレクタ１１２の出力と、後述するプロセッシングユニット３２から与えられるデータとの一方を選択してＡＬＵ７０の他方の入力に与えるためのセレクタ１３２とを含む。プロセッシングユニット３０はさらに、ＡＬＵ７０の出力の一方とレジスタファイル５８の出力とのいずれか一方を選択してＭＰＹ８０の一方の入力に与えるためのセレクタ１５０と、プロセッシングユニット３２から与えられるデータと、ＡＤＤ９０の出力とのいずれか一方を選択してＡＤＤ９０の一方の入力に与えるためのセレクタ１６２とを含む。プロセッシングユニット３０はさらにセレクタ１６０を含んでおり、このセレクタ１６０はＭＰＹ８０の出力とＡＤＤ９０の出力とのいずれか一方を選択してデータバス５２に出力するためのものである。

【０００６】プロセッシングユニット３２は、同様にセレクタ１１４、１１６、１３４、１３６、１５２、１６４を含む。プロセッシングユニット３４は、セレクタ１１８、１２０、１３８、１４０、１５４、１６６、１６８を含む。プロセッシングユニット３６は、セレクタ１２２、１２４、１４２、１４４、１５６、１７０を含む。さらに、セレクタ１６０と同様のセレクタが各プロセッシングユニット３２、３４、３６に含まれているが、図の簡略化のため図１７には示していない。

【０００７】セレクタ１１４、１１８、１２２は、セレクタ１１０と同様の機能を有する。セレクタ１１６、１２０、１２４は、セレクタ１１２と同様の機能を有する。セレクタ１３４、１３８、１４２は、セレクタ１３０と同様の機能を有する。セレクタ１３６は、セレクタ１１６の出力とセレクタ１１２の出力との一方を選択してＡＬＵ７２に与えるためのものである。セレクタ１４０はセレクタ１３２と同様である。セレクタ１４４は、セレクタ１２４の出力とセレクタ１２０の出力との一方をＡＬＵ７６に与えるためのものである。セレクタ１５２、１５４、１５６は、セレクタ１５０と同様の機能を有する。セレクタ１６４は、プロセッシングユニット３４の出力と乗算器８２の出力とのいずれか一方を選択して加算器９２に与えるためのものである。セレクタ１６６は、プロセッシングユニット３０の出力と、加算器９４の出力とのいずれか一方を選択して加算器９４に与えるためのものである。セレクタ１６８は、乗算器８４の出力と、プロセッシングユニット３６の出力とのいずれか一方を選択して加算器９４に与えるためのものである。セレクタ１７０は、乗算器８４の出力と、加算器９６の出力とのいずれか一方を選択して加算器９６に与えるためのものである。

【０００８】各プロセッシングユニット３０、３２、３４、３６には、各プロセッシングユニットを制御するためのローカル命令メモリＬＰＭ００、０１、１０、１１（図１７中では参照符号１００、１０２、１０４、１０６で示される）が設けられている。

【０００９】アドレス演算ユニット４８は、各メモリ３８、４０、４２、４４、４６の読出、書込アドレスを演算するためのものである。

【００１０】図１６および図１７に示されるごとく、従来のプロセッサにおいては、プロセッシングユニットの構成は相互に異なっており、相互の間の接続も、処理対象となる演算に合わせて特殊な形態となっている。

【００１１】図１８および図１９を参照して、従来のプロセッサは次のように動作する。プロセッシングユニット間のバス接続は、図１８と図１９とに示される２種類の構成から選択することができる。図１８に示される例においては、プロセッシングユニット３０、３２、３４、３６の間でのデータバス接続が存在しないように各セレクタが設定される。各プロセッシングユニットでは、積和演算が行なわれる。

【００１２】図１９に示される例では、プロセッシングユニット３２の乗算器８２の出力がプロセッシングユニット３０の加算器９０の入力に与えられる。加算器９０の出力は、プロセッシングユニット３４の加算器９４の入力に与えられる。一方プロセッシングユニット３４の乗算器８４の出力がプロセッシングユニット３６の加算器９６の入力の一方に与えられる。加算器９６の出力は

プロセッシングユニット３４の加算器１９４の入力の他方に与えられる。加算器９４の出力はプロセッシングユニット３２の加算器９２の入力の一方に与えられる。この図１９に示される接続では、４項ごとの積和演算が可能である。４項ごとの積和結果はプロセッシングユニット３２の出力として得られる。

【００１３】このプロセッサの制御は、セットアップ命令１つと、各プロセッシングユニットの制御を行なうための４個の命令との、合計５個の命令を単位として行なわれる。各命令は３２ビットであり、５個の命令では１６０ビットとなる。

【００１４】セットアップ命令は、各プロセッシングユニット入力部の５：１セレクタ１１０、１１２、１１４、１１６、１１８、１２０、１２２、１２４や、各プロセッシングユニット間のデータバスの接続を設定するためのセレクタなどを制御する。プロセッシングユニット制御命令は、メモリ３８、４０、４２、４４、４６のアドレスを発生したり、ローカル命令メモリ１００、１０２、１０４、１０６のアドレス指定を行なったりする。ローカル命令メモリ１００、１０２、１０４、１０６に含まれるローカル命令は、演算器で行なう演算内容を指定するためのものである。

【００１５】

【発明が解決しようとする課題】図１６～図１９に示される従来技術のプロセッサには、次のような問題点がある。このプロセッサは、もともと動画像圧縮用に積和演算の効率化を目標として開発された。そのため、同じように大量の演算が要求される処理であっても、動画像圧縮以外の分野へこのプロセッサを適用することは困難である。データの大量処理が要求される演算としては、積和演算のほかにもＦＦＴ（高速フーリエ変換）に用いられるバタフライ演算や、科学技術計算における倍精度演算などがある。バタフライ演算には、乗算器４個と加算器６個とが必要である。倍精度乗算（２ｎビットとする）では、ｎ×ｎビットの乗算器４個と２ｎ＋２ｎビットの加算器３個とが必要である。しかし、図１６～図１９に示されるプロセッサが行なえる処理は、動画像圧縮用の処理だけであり、そのハードウェアも、アルゴリズムが固定したものとして実現されている。これは、複数個のプロセッシングユニットを備えたプロセッサにおいて、上述のような様々な処理を行なおうとするとその制御が複雑になるなどの理由によるものである。したがって従来のこの種のプロセッサで汎用できるものは極めて少数であり、しかもその制御が複雑であったり、ハードウェアが複雑であるという欠点がある。

【００１６】この発明は上述の問題点に鑑みてなされたものであって、複数個のプロセッシングユニットを備えることにより並列処理を効率よく行なえるとともに、幅広い種類の演算を、比較的単純な制御方法で可能とする並列処理プロセッサとそのためのプロセッシングユニットと、プロセッサの動作方法とを提供することを目的とする。

【００１７】

【課題を解決するための手段】請求項１に記載の並列処理のためのプロセッシングユニットは、それぞれ複数個の入力を有し、与えられるデータの間に所定の演算を行なって結果を出力するための複数個の演算手段と、複数の単方向入力データバスに接続され、複数個の演算手段の各入力ごとに、複数の単方向入力データバスのうちのいずれか１つを可制御的に選択して、選択された単方向入力データバスを介して与えられるデータの一部を演算手段の入力に与えるための入力データバス選択手段と、複数個の演算手段の出力に接続された入力と、単方向入力データバスと同じ数の単方向出力データバスに接続された出力とを有し、演算手段の出力を、単方向出力データバスのいずれかに出力するための出力データバス選択手段と、複数個の演算手段により所望の複合演算を実現するために、入力データバス選択手段と、出力データバス選択手段とによるデータの経路を制御するための制御手段とを含む。

【００１８】請求項２に記載のプロセッシングユニットは、請求項１に記載のものであって、その複数個の演算手段が、２つのｎビット幅の入力を有し、与えられる２つのデータを乗算して２ｎビットの結果を出力する乗算器と、各々が２つのｎビット幅の入力を有し、与えられる２つのデータを加算してｎビット幅の結果を出力する２つの加算器とを含む。

【００１９】請求項３に記載のプロセッシングユニットは、請求項２に記載のものであって、複数の単方向入力データバスおよび複数の単方向出力データバスの各々は２ｎビット幅を有し、入力データバス選択手段は、複数個の演算手段の各入力ごとに複数の単方向入力データバスのうちのいずれか１つを可制御的に選択して、選択された単方向入力データバスを介して与えられるデータの上位または下位のｎビットを該入力に与えるための手段を含み、出力データバス選択手段は、演算手段の出力の各々を、単方向出力データバスの任意のいずれかの上位ｎビットまたは下位ｎビットまたはその双方に出力可能とするための手段を含む。

【００２０】請求項４に記載のプロセッシングユニットは、請求項２に記載のものであって、２つの加算器の一方はキャリー出力を有し、他方はキャリー入力を有し、さらに、キャリー出力とキャリー入力とを可制御的に断続するための手段を含む。

【００２１】請求項５に記載のプロセッシングユニットは請求項２に記載のものであって、入力データバス選択手段が、２つの加算器の１つの少なくとも１つの入力について、単方向入力データバスのうちのいずれか１つまたは該加算器自身の出力のいずれかを可制御的に選択して、選択された単方向入力データバスを介して与えられ

るデータまたは該加算器自身の出力のいずれかの一部を該入力に与えるための手段を含む。

【００２２】請求項６に記載のプロセッシングユニットは、請求項２に記載のものであって、入力データバス選択手段が、２つの加算器の１つの少なくとも１つの入力について、複数の単方向入力データバスのうちのいずれか１つまたは乗算器の出力のいずれかを可制御的に選択して、選択された単方向入力データバスを介して与えられるデータまたは乗算器の出力のいずれかの一部を該入力に与えるための手段を含む。

【００２３】請求項７に記載のプロセッシングユニットは、請求項２に記載のものであって、入力データバス選択手段が、２つの加算器の１つの少なくとも１つの入力について、複数の単方向入力データバスのうちのいずれか１つまたは該加算器自身の出力または乗算器の出力の一部のいずれかを可制御的に選択して、選択された単方向入力データバスを介して与えられるデータの一部または該加算器自身の出力または乗算器の出力の一部のいずれかを該入力に与えるための手段を含む。

【００２４】請求項８に記載のプロセッシングユニットは、請求項２に記載のものであって、所定の情報を予め記憶するための読出専用記憶手段をさらに含む。入力データバス選択手段は、乗算器の少なくとも１つの入力について、単方向入力データバスのうちのいずれか１つまたは読出専用記憶手段の出力のいずれかを可制御的に選択して、選択された単方向入力データバスを介して与えられるデータまたは読出専用記憶手段の出力のいずれかの一部を該入力に与えるための手段を含む。

【００２５】請求項９に記載の並列処理プロセッサは、ｎ個のプロセッシングユニットと、隣り合うプロセッシングユニットを所定方向に円環状に順次に接続するための第１の単方向データバスと、１つおいて隣り合うプロセッシングユニットを円環状に順次に双方向に接続するための、第２の単方向データバスとを含む。ｎは４のベキ乗である。各プロセッシングユニットは、それぞれ複数個の入力を有し、与えられるデータの間に所定の演算を行なって結果を出力するための複数個の演算手段と、隣接するプロセッシングユニットからの入力となる第１の単方向データバスと、第２の単方向データバスのうちの該プロセッシングユニットへの入力データバスとに接続され、複数個の演算手段の各入力ごとに第１および第２の単方向データバスのうちのいずれか１つを可制御的に選択して、選択された単方向データバスを介して与えられるデータの一部を該入力に与えるための入力データバス選択手段と、複数個の演算手段の出力に接続された入力と、隣接するプロセッシングユニットへの出力となる第１の単方向データバスと、第２の単方向データバスのうちの該プロセッシングユニットからの出力データバスとに接続された出力とを有し、演算手段の出力の各々を、第１および第２の単方向データバスのいずれかに出力するための出力データバス選択手段と、複数個の演算手段により所望の複合演算を実現するために、入力データバス選択手段と、出力データバス選択手段とによるデータの経路と、演算手段による演算の実行とを制御するための制御手段とを含む。

【００２６】請求項１０に記載の並列処理プロセッサは、請求項９に記載のものであって、プロセッシングユニットと同数の、各々が一度に２つのデータを出力可能なデータ記憶手段と、データ記憶手段の各々と、プロセッシングユニットの各々とを接続するための複数の読出データバスと複数の書込データバスとをさらに含む。各プロセッシングユニットにおいて、入力データバス選択手段は、隣接するプロセッシングユニットからの入力となる第１の単方向データバスと、第２の単方向データバスのうちの該プロセッシングユニットへの入力データバスと、複数の読出データバスとに接続され、複数個の演算手段の各入力ごとに第１および第２の単方向データバスと読出データバスとのうちのいずれか１つを可制御的に選択して、選択されたデータバスを介して与えられるデータの一部を入力に与えるための手段を含む。出力データバス選択手段は、複数個の演算手段の出力に接続された入力と、隣接するプロセッシングユニットへの出力となる第１の単方向データバスと、第２の単方向データバスのうちの該プロセッシングユニットからの出力データバスと、書込データバスとに接続された出力とを有し、演算手段の出力をこれらデータバスのいずれかに出力するための手段とを含む。

【００２７】請求項１１に記載のプロセッサは、請求項９に記載のものであって、プロセッシングユニットと同数の、各々が一度に２つのデータを出力可能なデータ記憶手段と、データ記憶手段の各々と、プロセッシングユニットの各々とを接続するための複数の読出データバスと複数の書込データバスとをさらに含む。複数のプロセッシングユニットは、各々が４のベキ乗個のプロセッシングユニットを含む複数個のグループに分割されており、複数のプロセッシングユニットと複数のデータ記憶手段とは１対１に対応付けられている。各プロセッシングユニットにおいて、入力データバス選択手段は、隣接するプロセッシングユニットからの入力となる第１の単方向データバスと、第２の単方向データバスのうちの該プロセッシングユニットへの入力データバスと、複数のデータバスのうち該プロセッシングユニットが含まれるグループのプロセッシングユニットと対応付けられたデータ記憶手段からの読出データバスとに接続され、複数個の演算手段の各入力ごとに第１および第２の単方向データバスと読出データバスとのうちのいずれか１つを可制御的に選択して、選択されたデータバスを介して与えられるデータの一部を該入力に与えるための手段を含む。出力データバス選択手段は、複数個の演算手段の出力に接続された入力と、隣接するプロセッシングユニッ

トへの出力となる第１の単方向データバスと、第２の単方向データバスのうちの該プロセッシングユニットからの出力データバスと、書込データバスのすべてとに接続された出力とを有し、演算手段の出力を、データバスのいずれかに出力するための手段を含む。

【００２８】請求項１２に記載の並列処理プロセッサの動作方法は、４個のプロセッシングユニットと、隣り合うプロセッシングユニットを所定方向に円環状に順次に接続するための４本の第１の単方向データバスと、１つおいて隣り合うプロセッシングユニットを双方向に接続するための、４本の第２の単方向データバスとを含む並列処理プロセッサの動作方法である。各プロセッシングユニットは、各々ｎビットの２つの入力を有し、与えられるデータの間に乗算を行なって２ｎビット幅の結果を出力するための乗算手段と、各々が、各々ｎビットの２つの入力を有し、与えられるデータの間に加算を行なってｎビット幅の結果を出力するための第１および第２の加算手段と、第１の加算手段のキャリー出力を第２の加算手段のキャリー入力に可制御的に与えるためのキャリー切換手段と、隣接するプロセッシングユニットからの入力となる第１の単方向データバスと、第２の単方向データバスのうちの該プロセッシングユニットへの入力データバスとなるものとに接続され、乗算手段と加算手段との各入力ごとに第１および第２の単方向データバスのうちのいずれか１つを可制御的に選択して、選択された単方向データバスを介して与えられるデータの一部を乗算手段と加算手段との入力にそれぞれ与えるための入力データバス選択手段と、乗算手段および加算手段の出力に接続された入力と、隣接するプロセッシングユニットへの出力となる第１の単方向データバスと、第２の単方向データバスのうちの該プロセッシングユニットからの出力データバスとに接続された出力とを有し、乗算手段および加算手段の出力を、第１および第２の単方向データバスのいずれかに出力するための出力データバス選択手段と、乗算手段および加算手段により所望の複合演算を実現するために、入力データバス選択手段と、出力データバス選択手段とによるデータの経路を制御するための制御手段とを含む。この動作方法は演算に必要なデータの各々を入力データバス選択手段に与えるステップと、入力データバス選択手段により、データの各々を上位および下位のｎビットずつに分解し、４つのプロセッシングユニットの、乗算手段および加算手段の入力のいずれか２つにそれぞれ与えるステップと、すべてのプロセッシングユニットのキャリー切換手段を、所望の演算に応じて設定するステップと、各プロセッシングユニットの出力データバス選択手段および入力データバス選択手段を制御して、所望の演算が得られるように各プロセッシングユニットの乗算手段と、第１および第２の加算手段との間の接続を設定するステップと、演算結果が、データバスのうちの所望のものに出力されるように、所望の演算によって定まる所定のプロセッシングユニットの乗算手段および加算手段のうちの所定のものの出力の、所定の部分を所望のデータバスに出力するように出力データバス選択手段を制御するステップとを含む。

【００２９】請求項１３に記載の動作方法は、請求項１２に記載のものであって、接続を設定するステップは、あるプロセッシングユニット内の乗算手段または加算手段の出力の上位ｎビットが、他のプロセッシングユニット内の乗算手段または加算手段の下位ｎビットに入力されるように、これらプロセッシングユニットの出力データバス選択手段および入力データバス選択手段によるデータ経路を設定するステップを含む。

【００３０】請求項１４に記載の動作方法は、請求項１２に記載のものであって、接続を設定するステップは、あるプロセッシングユニット内の乗算手段または加算手段の出力の上位ｎビットが、他のプロセッシングユニットへのデータバスの下位ｎビットに出力されるように、該プロセッシングユニットの出力データバス選択手段によるデータ経路を設定するステップを含む。

【００３１】請求項１５に記載の並列処理プロセッサ、請求項９に記載のものであって、各プロセッシングユニットごとに準備された、制御手段が実行する制御命令を格納するための命令記憶手段をさらに含む。

【００３２】請求項１６に記載の並列処理プロセッサは、請求項１５に記載のものであって、制御命令は、データ記憶手段を制御するための第１の種類の制御命令と、制御手段による演算手段の制御のための第２の種類の制御命令との２つの系統に分類される。

【００３３】請求項１７に記載の並列処理プロセッサは、請求項１６に記載のものであって、第１の種類の制御命令は、データ記憶手段の各々の２つの読出アドレスと、１つの書込アドレスとを指定する。

【００３４】請求項１８に記載の並列処理プロセッサは、請求項１６に記載のものであって、制御手段は、与えられる第２の種類の制御命令が変更されるまでは、直前に与えられた第２の種類の制御命令に従って演算手段と入力データバス選択手段と出力データバス選択手段とを制御する。

【００３５】請求項１９に記載の並列処理プロセッサは、請求項１６に記載のものであって、命令記憶手段は、複数個の命令を記憶する命令メモリと、命令メモリの読出アドレスを指定するためのプログラムカウンタと、プログラムカウンタにより指定されたアドレスを先頭として２つの命令を一度に読出すための手段と、読出された２つの命令が同一の系統に属するか否かを判断するための手段と、判断結果に従って、制御手段またはデータ記憶手段またはその双方に制御命令を与えるための手段と、判断結果に従って、プログラムカウンタのカウントを１または２増加させるための手段とを含む。

【００３６】

15

【作用】請求項１に記載の並列処理のためのプロセッシングユニットにおいては、そこに接続される単方向入力データバスの数と、単方向出力データバスの数とが同じである。そしてこれら単方向入力データバスの任意のものを介して与えられるデータを、複数個の演算手段で処理して結果を単方向出力データバスの任意のものに出力できる。制御手段の制御により種々の演算が可能なため、このプロセッシングユニットは汎用的であり、かつ制御手段の制御は比較的単純でよい。しかも入出力データバスの数が同一であるために、このプロセッシングユニットを複数個組合わせて並列処理プロセッサを作製するときのプロセッシングユニット相互の接続が容易であり、各プロセッシングユニット構成が同一であるために、それらの制御に互換性を持たせることができる。

【００３７】請求項２に記載のプロセッシングユニットは、ｎ×ｎビットの乗算器とｎ＋ｎビットの２つの加算器とを用いて、大量のデータ処理が要求される演算における典型的な処理を行なうことができる。

【００３８】請求項３に記載のプロセッシングユニットでは、演算手段の各入力に、単方向入力データバスのうちのいずれか任意の１つの上位または下位のｎビットが与えられる。これら演算手段の出力の各々は、単方向出力データバスの任意のいずれかの上位ｎビットまたは下位ｎビットたはその双方に出力される。したがって２ｎビットのデータ同士の演算を、それぞれ上位ｎビットと下位ｎビットとに分けてｎ×ｎビットの乗算器およびｎ＋ｎビットの加算器を用いて行なうことができる。

【００３９】請求項４に記載のプロセッシングユニットでは、２つの加算器の一方のキャリー出力を他方のキャリー入力とする場合と、しない場合とを選択できる。したがって、２ｎビットのデータ同士の加算と、ｎビットのデータ同士の加算と、ｎビットのデータ同士の２つの別個の加算とを実行することができる。

【００４０】請求項５に記載のプロセッシングユニットでは、２つの加算器の１つの少なくとも１つの入力について、入力データとして複数の単方向入力データバスのうちのいずれか１つまたは該加算器自身の出力のいずれかを選択できるので、自己の加算結果を用いる演算、たとえば積和演算と、データバス経由のデータを用いた加算との双方を行なうことができる。

【００４１】請求項６に記載のプロセッシングユニットでは、２つの加算器の１つの少なくとも１つの入力について、入力データとして単方向入力データバスのうちのいずれか１つまたは乗算器の出力のいずれかを選択できるので、乗算結果を加算の入力に用いる演算、たとえば積和演算と、データバス経由のデータを用いた演算との双方を行なうことができる。

【００４２】請求項７に記載のプロセッシングユニットでは、２つの加算器の１つの少なくとも１つの入力について、入力データとして単方向入力データバスのうちの

16

いずれか１つと、該加算器自身の出力と、乗算器の出力の一部とのいずれかを選択できる。したがって、乗算結果や加算結果を加算の入力に用いる演算、たとえば積和演算と、データバス経由のデータを用いた加算との双方を行なうことができる。

【００４３】請求項８に記載のプロセッシングユニットでは、乗算器の少なくとも１つの入力について、入力データとして単方向入力データバスのうちのいずれか１つと読出専用記憶手段の出力とのいずれかを選択できる。したがって、読出専用記憶手段に格納されたデータを用いることにより処理の高速化を図れる処理、たとえばニュートン・ラプソン法による除算や、開平演算などを効率よく行なうための構成と、通常の演算を行なうための構成とを自由に選択できる。

【００４４】請求項９に記載の並列処理プロセッサでは、ｎ個のプロセッシングユニットが、隣り合うプロセッシングユニットとは第１の単方向データバスにより単方向に、１つおいて隣り合うプロセッシングユニットとは第２の単方向データバスにより双方向に、それぞれ接続される。各プロセッシングユニットは、最低でも２以上の同じ数の入力と出力とを有する。入出力の数が多く、演算の自由度が高くなる。また、各プロセッシングユニットは同一の構造を有するため、プロセッサのレイアウトおよび制御の複雑さが低減される。

【００４５】請求項１０に記載の並列処理プロセッサでは、各プロセッシングユニットごとに、データ記憶手段が準備される。各プロセッシングユニットは、データ記憶手段からのデータと、他のプロセッサからのデータとのいずれにも、複数個の演算手段を用いた所定の処理を行ない、任意の演算手段の出力を他のプロセッシングユニットと、データ記憶手段との任意のものに出力できる。

【００４６】請求項１１に記載の並列処理プロセッサでは、プロセッシングユニットはグループに分類され、各プロセッシングユニットごとに、データ記憶手段が準備される。各プロセッシングユニットは、同一のグループに属するプロセッシングユニットに対応するデータ記憶手段からのデータと、他のプロセッサからのデータとのいずれにも、複数個の演算手段を用いた所定の処理を行ない、任意の演算手段の出力を他のプロセッシングユニットと、データ記憶手段との任意のものに出力できる。データ記憶手段から、同じグループに属するプロセッシングユニットへの読出データバスは、プロセッシングユニットをグループ化せず全体に接続する場合と比較して短くてすみ、プロセッシングユニットへのデータの供給が高速に行なえる。また１グループのプロセッシングユニットで共通のデータを用いた演算を実行できる。さらに、各プロセッシングユニットからは任意のデータ記憶手段にデータを書込めるので、グループ間のデータ交換も可能である。

17

【００４７】請求項１２に記載の並列処理プロセッサの動作方法では、データの各々が上位および下位のｎビットずつに分解され、４つのプロセッシングユニットの、乗算手段および加算手段の入力のいずれか２つにそれぞれ与えられる。他のプロセッシングユニットの演算結果の上位ｎビットまたは下位ｎビットは、新たな演算のためのデータの下位ｎビットまたは上位ｎビットとして乗算手段または加算手段に与えることができ、多様な演算を行なうことができる。また、２つの加算手段の間でキャリーの入出力を行なうかどうかを選択できるので、ｎビット精度の２つの加算処理と２ｎビット精度の１つの加算処理とを切換えて行なうことができる。したがって、非常に自由度の高い処理を行なうことができる。

【００４８】請求項１３に記載の並列処理プロセッサの動作方法では、あるプロセッシングユニット内の乗算手段の出力の上位ｎビットまたは下位のｎビットが、他のプロセッシングユニット内の乗算手段または加算手段の下位ｎビットと上位ｎビットとの任意の一方に入力されるようにデータバスと乗算手段または加算手段との接続を設定できる。シフト手段を用いることなく多様な演算を行なうことが可能である。

【００４９】請求項１４に記載の並列処理プロセッサの動作方法では、あるプロセッシングユニット内の乗算手段の出力の上位ｎビットが、他のプロセッシングユニット内の乗算手段または加算手段の下位ｎビットに入力されるようにデータバスと乗算手段または加算手段との接続を設定できる。シフト手段を用いることなくデータを実質的にシフトでき、シフトのための時間も必要としない。

【００５０】請求項１５に記載の並列処理プロセッサでは、各プロセッシングユニットごとに制御命令が命令記憶手段に格納される。複雑な演算を各プロセッシングユニットごとに制御すればよく、並列処理プロセッサ全体の制御が容易になる。

【００５１】請求項１６に記載の並列処理プロセッサでは、制御命令は、データ記憶手段を制御するための制御命令と、演算手段の制御のための制御命令との２つの系統に分類される。データ記憶手段から出力されるデータを介しつつ一定の命令を繰り返し行なう場合などに、データ記憶手段を制御する制御命令のみを、その内容を変化させつつ繰り返し発行すれば、演算手段のための異なる制御命令をいくつも発行する必要がない。

【００５２】請求項１７に記載の並列処理プロセッサでは、データ記憶手段の各々の２つの読出アドレスからの２つのデータを読出して所定の演算を行ない、結果を１つの書込アドレスにより指定された位置に書込できる。

【００５３】請求項１８に記載の並列処理プロセッサでは、第２の種類の制御命令が変更されるまでは、直前に与えられた第２の種類の制御命令に従って演算手段と入力データバス選択手段と出力データバス選択手段とが制

18

御される。データ記憶手段から出力されるデータを介しつつ一定の命令を繰り返し行なう場合などに、データ記憶手段を制御する命令のみを、その内容を変化させつつ繰り返し発行すればよく、演算手段のための同一の制御命令を繰り返し発行する必要がない。

【００５４】請求項１９に記載の並列処理プロセッサでは、命令メモリから、プログラムカウンタにより指定されたアドレスを先頭として２つの命令が一度に読出され、読出された２つの命令が同一の系統に属するか否かが判断される。同一であれば先に読出出された命令のみ実行し、プログラムカウンタを１増加させる。同一でなければ制御手段とデータ記憶手段との双方にそれぞれの制御命令を与え、プログラムカウンタを２増加させる。異なる系統の命令は同時に実行可能なので、一度に制御命令を１つしか読出さない場合と比較して、命令の実行速度が向上する。

【００５５】

【実施例】

［第１の実施例］本発明の一実施例を、４個のプロセッシングユニット（ＰＵ）から構成されるプロセッサについて以下に示す。図１を参照してこのプロセッサは、４つのプロセッシングユニットＰＵ００、０１、１０、１１（図１中ではそれぞれ参照符号２００、２０２、２０４、２０６により示される）と、これらプロセッシングユニット２００、２０２、２０４、２０６を、この順序で円環状に接続するための単方向データバス２１０、２１２、２１４、２１６とを含む。プロセッシングユニットＰＵに付けられた数値（００、０１、１０、１１）は、そのプロセッシングユニットのアドレスを２進数で表わしたものである。

【００５６】このプロセッサはさらに、アドレスが２だけ異なるプロセッシングユニットを双方向に接続するための、単方向入力データバス２２０、２２２、２２４、２２６を含む。前述の単方向データバス２１０、２１２、２１４は、アドレスが１だけ異なるプロセッシングユニットを順次単方向に接続するためのものである。またデータバス２１６は、プロセッシングユニットＰＵ１１（２０６）からＰＵ００（２００）へのデータバスである。図１において、各データバスに付加された矢印は、データの流れる方向を表わしている。図１に示されるプロセッサでは、各プロセッシングユニットに入るデータバスの入力数と、各プロセッシングユニットから出るデータバスの出力数とは相互に等しく、それぞれ２となっている。すべてのプロセッシングユニットにおいてこの数は等しい。

【００５７】図２を参照して、この第１の実施例のプロセッサは、前述のプロセッシングユニットＰＵ００、０１、１０、１１（２００、２０２、２０４、２０６）と、データバス２１０、２１２、２１４、２１６、２２０、２２２、２２４、２２６とに加えて、各プロセッサ

200、202、204、206に対応してそれぞれ設けられたコントロール回路（PUC）250、252、254、256と、命令メモリ（IM）260、262、264、266と、データメモリ270、272、274、276とを含む。

【0058】このプロセッサはさらに、データメモリ270、272、274、276から読出されるデータためのメモリ読出バス群280と、各プロセッシングユニット200、202、204、206からデータメモリ270、272、274、276に書込むデータのためのメモリ書込バス群272とを含む。各データメモリ270、272、274、276はそれぞれ同時に2個のデータの読出が可能であり、それぞれデータバス300と301、302と303、304と305、306と307によりメモリ読出バス群280内の互いに異なるデータバスに接続されている。この接続については後述する。メモリ読出データバス群280内のデータバスは、データバス群310、312、314、316によりプロセッシングユニット200、202、204、206にそれぞれ接続される。

【0059】メモリ読出データバス群282は、4つのnビットデータバスを含み、同じくそれぞれ4本ずつのnビット幅のデータバスからなるデータバス群320、322、324、326によりプロセッシングユニット200、202、204、206の出力に接続されている。この接続の詳細については後述する。データメモリ270、272、274、276へのデータの書込は同時に1個が可能である。データメモリに書込むデータは各プロセッシングユニットの演算結果である。

【0060】命令メモリ260、262、264、266へは、図示されない入出力ポートを介して外部より与えられる命令が格納される。コントロール回路250、252、254、256の各々は、2個の出力を持ち、出力の一方はデータメモリ270、272、274、276の制御に、他方はプロセッシングユニット200、202、204、206内の演算器制御にそれぞれ用いる。

【0061】図3に、プロセッシングユニット200の内部構成を示す。図3にはプロセッシングユニット200の構成を例として示すが、他のプロセッシングユニット202、204、206の構成もこのプロセッシングユニット200と全く同一である。したがって、ここではそれらについての詳しい説明は繰り返さないこととする。

【0062】図3を参照して、メモリ書込バス群282は、4本のメモリ書込バス400、402、404、406を含む。また図1および図2に示されるプロセッシングユニット202へのデータバス210は、上位nビットのデータバス210Mと下位nビットのデータバス210Lとを含む。同様にプロセッシングユニット204へのデータバス220は、上位nビットのデータバス220Mと下位nビットのデータバス220Lとを含む。

【0063】また図2に示されるメモリ読出バス群280からプロセッシングユニット200へのメモリ読出バス群310は、図3に示されるようにデータバス330、332、334、336、340、342、344、346を含む。前述のようにメモリ読出バス群280は8本のデータバスを含み、これらメモリ読出バス330、332、334、336、340、342、344、346はその8本のデータバスからそれぞれ分岐したものである。この接続については図4を参照して後述する。

【0064】図1および図2に示されるプロセッシングユニット204からのデータバス224は、上位nビットのデータバス224Mと下位nビットのデータバス224Lとを含む。またプロセッシングユニット206からのデータバス216も同様に、上位nビットのデータバス216Mと下位nビットのデータバス216Lとを含む。

【0065】図3を参照して、このプロセッシングユニット200は、読出バス群310に接続されたセレクタ350、352、354、356と、セレクタ350、352、354、356の出力とデータバス216、224となどが入力に接続されたセレクタ360、362、364、366、368、370と、セレクタ360と362との出力が入力に接続された乗算器（MPY）380と、セレクタ368と370との出力に接続された2つの入力を有する第1の加算器（ADD0）384と、セレクタ364と366との出力に接続された2つの入力を有する第2の加算器（ADD1）382と、乗算器380のそれぞれnビット幅の出力MPMとMPLと、加算器384、382の出力とに接続され、各出力データを出力データバス210、220の上位nビットのデータバス210Mおよび220Mと、下位nビットのデータバス210Lおよび220Lとのいずれかに出力するためのクロスバースイッチ（CBS）392と、乗算器380の2つの出力MPM、MPLと、2つの加算器382、384の出力とを、メモリ書込バス群282の4つのメモリ書込バス400、402、404、406のいずれかに出力するためのセレクタ（SELW）390とを含む。セレクタ390とメモリ書込バス400、402、404、406とはそれぞれデータバス410、412、414、416により接続されている。クロスバースイッチ392は、それぞれ2nビットのデータバス420、422によりデータバス210、220に接続されている。データバス420の上位nビットがデータバス210Mに接続され、下位nビットがデータバス210Lに接続されている。データバス422の上位nビットがデータバス220Mに接続さ

れ、下位ｎビットがデータバス２２０Ｌに接続されている。

【００６６】セレクタ３５０および３５４の入力には、４組のデータバス３３０、３３２、３３４、３３６がそれぞれ接続されている。セレクタ３５２、３５６の入力には、４組のデータバス３４０、３４２、３４４、３４６がそれぞれ接続されている。

【００６７】セレクタ３６０の一方の入力にはセレクタ３５０の出力が接続されている。セレクタ３６０の他方の入力には、データバス２２４の上位ｎビットのデータバス２２４Ｍが接続されている。セレクタ３６２の一方の入力にはセレクタ３５２の出力が接続されている。セレクタ３６２の他方の入力には、データバス２１６の上位ｎビット２１６Ｍが接続されている。

【００６８】セレクタ３６４、３６６はそれぞれ３入力である。セレクタ３６４の１つの入力には、セレクタ３５４の出力が接続されている。セレクタ３６４の他の１つの入力には、データバス２２４の上位ｎビットのデータバス２２４Ｍが接続されている。セレクタ３６４の残りの１つの入力には、加算器３８２の出力ＡＤ１が接続されている。セレクタ３６６の入力の１つにはセレクタ３５６の出力が接続されている。セレクタ３６６の他の入力の１つには、乗算器３８０の出力のうちの上位ｎビットＭＰＭが与えられる。セレクタ３６６の残りの１つの入力には、データバス２１６の上位ｎビットのデータバス２１６Ｍが接続される。

【００６９】セレクタ３６８、３７０はそれぞれ４入力を有する。セレクタ３６８の第１の入力には、加算器３８４の出力ＡＤ０が与えられる。第２の入力には、セレクタ３５０の出力が与えられる。第３の入力にはデータバス２２４の下位ｎビットのデータバス２２４Ｌが接続される。第４の入力には、データバス２２４の上位ｎビットのデータバス２２４Ｍが接続される。セレクタ３７０の第１の入力は、セレクタ３５２の出力に接続される。第２の入力はデータバス２１６の下位ｎビットのデータバス２１６Ｌに接続される。第３の入力には乗算器ＭＰＹ３８０の出力の下位ｎビットＭＰＬが与えられる。第４の入力はセレクタ３５６の出力に接続される。

【００７０】加算器３８４のキャリー出力と加算器３８２のキャリー入力との間にはキャリー出力スイッチ３８６が設けられている。キャリー出力スイッチ３８６は、制御信号ＣＣにより制御されて開閉する。

【００７１】図４を参照して、メモリ書込バス群２８２の４本のデータバス４００、４０２、４０４、４０６は、それぞれデータメモリ２７０、２７２、２７４、２７６に接続される。一方メモリ読出バス群２８０は８本のメモリ読出バス２９０～２９７を含む。データメモリ２７０は、メモリ読出バス３００および３０１によりメモリ読出バス２９０、２９１に接続される。データメモリ２７２は、メモリ読出バス３０２、３０３によりメモリ読出バス２９２、２９３に接続される。データメモリ２７４はメモリ読出バス３０４、３０５によりメモリ読出バス２９４、２９５に接続される。データメモリ２７６はメモリ読出バス３０６、３０７によりメモリ読出バス２９６、２９７に接続される。メモリ読出バス２９０～２９７はそれぞれ分岐して、メモリ読出バス群３１０、３１２、３１４、３１６として図２に示されるプロセッシングユニット２００、２０２、２０４、２０６に接続されている。

【００７２】図３に示される構成を有するプロセッシングユニットにより、以下の演算が可能となる。

【００７３】（１）　データメモリから読出されたデータ同士の間でのｎ×ｎビット乗算、ｎ＋ｎビット加算。

【００７４】（２）　データメモリから読出されたデータと、データバス２２４Ｍまたは２１６Ｍ上から与えられるデータとの間のｎ×ｎビット乗算、ｎ＋ｎビット加算。

【００７５】（３）　データバス２２４Ｍと２１６Ｍ上のデータの間でのｎ×ｎビット乗算、ｎ＋ｎビット加算。

【００７６】（４）　乗算器３８０の出力する乗算結果の上位ｎビット（ＭＰＭ）と、データバス２２４Ｍ上のデータとの間の加算、および乗算器ＭＰＹの出力の下位ｎビット（ＭＰＬ）とデータバス２２４Ｌの上のデータの間の加算。

【００７７】（５）　乗算結果の上位ｎビット（ＭＰＭ）とデータメモリからのデータとの間の加算、乗算結果の下位ｎビット（ＭＰＬ）とデータメモリからのデータとの間の加算。

【００７８】（６）　乗算結果の上位ｎビット（ＭＰＭ）と加算器３８２の加算結果との間の加算、乗算結果の下位ｎビット（ＭＰＬ）と加算器３８４の加算結果との間の加算（積和演算）。

【００７９】（７）　データバス２２４Ｍと２２４Ｌとにより表現される２ｎビット数とデータバス２１６Ｍと２１６Ｌとにより表現される２ｎビット数との間の加算。

【００８０】図１～図４に示される第１の実施例の並列処理プロセッサにつき、制御方式を以下に説明する。この第１の実施例の並列処理プロセッサでは、各プロセッシングユニットが独立に制御される。各プロセッシングユニットに対応して命令メモリ２６０、２６２、２６４、２６６（図２参照）が備えられている。

【００８１】各プロセッシングユニットの命令は、データメモリを制御するデータメモリ制御系命令と、プロセッシングユニット内の演算器を制御する演算器制御系命令の少なくとも２系統に分類される。データメモリ制御系命令は、図２に示されるコントロール回路２５０、２５２、２５４、２５６から対応のデータメモリ２９０、２９２、２９４、２９６にそれぞれ接続されたバスに出

力されるものである。演算器制御系命令は、各コントロール回路２５０、２５２、２５４、２５６から、対応のプロセッシングユニット２００、２０２、２０４、２０６に向かうバスに出力される。

【００８２】図５（ａ）は、データメモリ制御系命令４３０の形式を示す。データメモリ制御系命令４３０は、ＯＰフィールド４３２と、ｓｒｃ０、ｓｒｃ１フィールド４３４、４３６と、ｄｓｔフィールド４３８とを含む。

【００８３】ＯＰフィールド４３２は、アドレスモードの指定を行なうためのものである。ｓｒｃ０、ｓｒｃ１フィールドは、対応のデータメモリから読出すデータの２つのアドレスを指定するためのものである。ｄｓｔフィールド４３８は、データメモリに書込むデータのアドレスおよび演算器出力とメモリ書込バス４００、４０２、４０４、４０６との間の接続を指定するためのものである。

【００８４】図５（ｂ）に、演算器制御系命令４５０の形式を示す。演算器制御系命令４５０は、乗算器を制御するためのＭＰＹフィールドと、加算器ＡＤＤ１、ＡＤＤ０を制御するためのＡＤＤ１フィールド、ＡＤＤ０フィールドと、８本のメモリ読出バス２９０～２９７から４個のデータを選択するためのデータを格納するＳＥＬフィールドとを含む。

【００８５】ＭＰＹフィールドは、ＯＰ０フィールド４５２と、ｓｒｃ００フィールド４５４と、ｓｒｃ０１フィールド４５６と、ｄｓｔ０フィールド４５８とを含む。ＡＤＤ１フィールドは、ＯＰ１フィールド４６０と、ｓｒｃ１０フィールド４６２と、ｓｒｃ１１フィールド４６４と、ｄｓｔ１フィールド４６６とを含む。ＡＤＤ０フィールドは、ＯＰ２フィールド４６８と、ｓｒｃ２０フィールド４７０と、ｓｒｃ２１フィールド４７２と、ｄｓｔ２フィールド４７４とを含む。

【００８６】ＯＰ０フィールド４５２と、ＯＰ１フィールド４６０と、ＯＰ２フィールド４６８とは、それぞれ対応の各演算器の演算内容を指定するためのものである。ｓｒｃ００フィールド４５４と、ｓｒｃ０１フィールド４５６と、ｓｒｃ１０フィールド４６２と、ｓｒｃ１１フィールド４６４と、ｓｒｃ２０フィールド４７０と、ｓｒｃ２１フィールド４７２とは、各演算器の入力に設けられたセレクタを制御するためのデータを格納する。ｄｓｔ０フィールド４５８と、ｄｓｔ１フィールド４６６と、ｄｓｔ２フィールド４７４とは、各演算器とデータバスを接続するためのクロスバースイッチＣＢＳ３９２（図３参照）を制御するためのデータを格納する。

【００８７】ＳＥＬフィールドは、ＳＥＬ０フィールド４７６と、ＳＥＬ１フィールド４７８と、ＳＥＬ２フィールド４８０と、ＳＥＬ３フィールド４８２とを含む。各フィールドは、それぞれ８本のメモリ読出バスから１個を選択するためのデータを格納する。したがってＳＥＬフィールドにより４つのデータが選択される。

【００８８】図５（ｂ）に示されるフィールドのうち、ｄｓｔ０フィールド４５８は、図示していないがさらに２個のフィールドｄｓｔ００フィールドとｄｓｔ０１フィールドとに分割され、それぞれ乗算器ＭＰＹの２つの出力ＭＰＭおよびＭＰＬの出力先を指定するためのデータを格納する。

【００８９】図５（ｂ）に示されるフィールド４５４、４５６、４５８（上述の２つのフィールドｄｓｔ００、ｄｓｔ０１）と、フィールド４６２、４６４、４６６、４７０、４７２、４７４とに格納されるデータの値と、各値に対応する各セレクタの選択動作とを、以下の第１表～第１０表に示す。

【００９０】

【表１】

［第１表］

ｓｒｃ００	入　力　信　号
0	メモリ
1	データバス２２４Ｍ

［第２表］

ｓｒｃ０１	入　力　信　号
0	メモリ
1	データバス２１６Ｍ

【００９１】

【表２】

25

［第３表］

ｄｓｔ００	出　　力　　先
００	データバス２１０Ｍ
０１	データバス２１０Ｌ
１０	データバス２２０Ｍ
１１	データバス２２０Ｌ

［第４表］

ｄｓｔ０１	出　　力　　先
００	データバス２１０Ｍ
０１	データバス２１０Ｌ
１０	データバス２２０Ｍ
１１	データバス２２０Ｌ

【００９２】
【表３】

26

［第５表］

ｓｒｃ００	入　力　信　号
００	ＡＤ１
０１	メモリ
１０	データバス２２４Ｍ
１１	－

［第６表］

ｓｒｃ１１	入　力　信　号
００	メモリ
０１	ＭＰＭ
１０	データバス２１６Ｍ
１１	－

［第７表］

ｄｓｔ１	出　　力　　先
００	データバス２１０Ｍ
０１	データバス２１０Ｌ
１０	データバス２２０Ｍ
１１	データバス２２０Ｌ

【００９３】
【表４】

27

［第８表］

ｓｒｃ２０	入　力　信　号
００	ＡＤ０
０１	メモリ
１０	データバス２２４Ｌ
１１	データバス２２４Ｍ

［第９表］

ｓｒｃ２１	入　力　信　号
００	メモリ
０１	ＭＰＭ
１０	データバス２１６Ｍ
１１	－

［第１０表］

ｄｓｔ２	出　力　先
００	データバス２１０Ｍ
０１	データバス２１０Ｌ
１０	データバス２２０Ｍ
１１	データバス２２０Ｌ

図５（ａ）に示されるｄｓｔフィールド４３８は、さらに３つのフィールドＰＵ選択フィールド４４０と、ＳＥＬＷ制御フィールド４４２と、書込アドレス指定フィールド４４４とに分割される。これらフィールドのうちフィールド４４０、４４２はいずれも２ビット長である。これらのフィールド４４０、４４２に格納されるデータの値と、各値により選択されるプロセッシングユニットと、ＳＥＬＷにより選択される各演算器の出力との一覧を次の第１１表および第１２表にそれぞれ示す。

【００９４】

【表５】

28

［第１１表］

ＰＵ選択フィールド	選択ＰＵ
００	ＰＵ００
０１	ＰＵ０１
１０	ＰＵ１０
１１	ＰＵ１１

［第１２表］

ＳＥＬＷ制御フィールド	選択出力
００	ＭＰＭ
０１	ＭＰＬ
１０	ＡＤ１
１１	ＡＤ０

図２に示されるコントロール回路２５０、２５２、２５４、２５６は、図５、図６に示される命令に従って、第１表～第１２表に示されるように各セレクタなどを制御する。

【００９５】図６に、図２に示される命令メモリ２６０、２６２、２６４、２６６への命令の格納方式を示す。図６において「ＭＣＮＴ」で示されるのはデータメモリ制御系命令であり、「ＰＣＮＴ」で示されるのは演算器制御系命令である。命令メモリ２６０は、基本的には命令４９０および４９２に示されるように、上述のデータメモリ制御系命令と演算器制御系命令とを組にして格納する。図６において命令メモリ２１６の左側に示す数字（１００、１０１、１０２、１０３）は命令メモリ２６０のアドレスを示す。図６に示される例では１００番地にＭＣＮＴ命令４９０が、１０１番地にＰＣＮＴ命令４９２がそれぞれ格納されている。プログラムカウンタアドレスが「１００」を指している場合には、次の命令として１００番地のＣＮＴ命令４９０が読出されることを示す。

【００９６】前述のように基本的には命令はデータメモリ制御系命令ＭＣＮＴと演算器制御系命令ＰＣＮＴとを組として取扱っている。しかし、大量のデータに同一の演算を繰り返す場合、各演算器への入力条件および演算内容は最初に一度だけ設定すればよく、データの位置を示すデータメモリのアドレスのみを順次変更していくことで処理できる。そのような場合には、図６のアドレス

29

１０２、１０３以下で示されるように、ＭＣＮＴ命令４９４、４９６を連続して命令メモリ２６０に格納しておく。各演算器は次の演算器制御系命令ＰＣＮＴを受取るまでは、前回に設定された演算器制御系命令に基づいて同じ演算内容を繰り返し実行する。

【００９７】以下、この第１の実施例の並列処理プロセッサの動作につき、具体例を用いて順次説明する。以下の例において、各プロセッシングユニット内のセレクタ、クロスバースイッチは、第１表～第１２表に従い、それぞれの図に示されるような接続を与えるように設定された命令で、予め所望の接続を与えるように切換えられているものとする。

【００９８】図７は、各プロセッシングユニット２００、２０２、２０４、２０６が、それぞれ独立にｎビット精度の演算を行なう例を示す。この場合には、プロセッシングユニット間を接続するデータバスは使用しない。各プロセッシングユニットＰＵ００～ＰＵ１０（２４０、２４２、２４４、２４６）において、「×」は乗算器を、「＋」は加算器をそれぞれ示す。

【００９９】プロセッシングユニット２００においては、乗算器への２つの入力はともにデータメモリからのデータである。乗算器はｎ×ｎビット構成であり、その出力は２ｎビットである。この例の場合には、乗算器の出力のうちの上位ｎビットまたは２ｎビットに丸め演算を行なった後の上位ｎビットをデータメモリに出力するように図３に示されるセレクタＳＥＬＷ３９０が設定されるものとする。丸め演算には専用のハードウェアが必要であるが、本願発明とは直接の関連がないため、その図示および説明はここでは行なわない。

【０１００】プロセッシングユニット２０２，２０４においては、それぞれの加算器の一方の２つの入力に、ともにデータメモリからのデータが与えられる。すなわち、各加算器の入力部分のセレクタが、データメモリからのデータを選択するように設定されている。プロセッシングユニット２０６では、積和演算が行なわれている。すなわち、乗算器の２つの入力に、データメモリからの２つのデータが与えられる。乗算器の出力の上位ｎビットが加算器の一方の入力に与えられ、加算器の出力がその加算器自身の他方の入力に与えられている。

【０１０１】図７に示される接続例では、各プロセッシングユニットからの出力は、メモリ書込バス４００、４０２、４０４、４０６（図３参照）を介してデータメモリに書込まれる。これは以下に示す他の接続例でも同様であり、所望の演算結果が得られる乗算器または加算器の出力が、メモリ書込バス４００、４０２、４０４、４０６のうちの所望のものに書込まれるように、各プロセッシングユニットのセレクタＳＥＬＷ３９０が制御されるものとする。

【０１０２】図８は、倍精度２ｎビットの乗算を行なう場合の、この実施例の並列処理プロセッサのデータバス

30

の接続関係を示す。乗算対象のデータをそれぞれａ、ｂとする。データａの上位ｎビットと下位ｎビットとをそれぞれａ１、ａ０と表わす。データｂの上位ｎビットと下位ｎビットとをそれぞれｂ１、ｂ０として表わす。すると乗算「ａ×ｂ」は次のように書ける。

【０１０３】（ａ０＋ａ１）×（ｂ０＋ｂ１）
＝ａ０×ｂ０
＋ａ０×ｂ１
＋ａ１×ｂ０
＋ａ１×ｂ１

すなわち、２ｎビット数同士の乗算ａ×ｂは、４個のｎビット数同士の乗算ａ０×ｂ０、ａ０×ｂ１、ａ１×ｂ０、ａ１×ｂ１を足し合せたものに分解できる。図８に示される接続例は、２ｎビットの２つの数ａ、ｂを上位ｎビット、下位ｎビットに分解して上述の計算を行なうためのものである。

【０１０４】以下、各プロセッシングユニットごとにその接続関係について説明する。なお、各プロセッシングユニット内の加算器のうち左側がＡＤＤ１、右側がＡＤＤ０である。

【０１０５】プロセッシングユニット２００では、乗算器の２つの入力にはデータメモリからの２つのデータが与えられるように各セレクタが設定される。乗算器の出力の２ｎビットのうち上位ｎビットが、データバス２１０Ｌを介してプロセッシングユニット２０２の加算器ＡＤＤ０の一方の入力に接続される。プロセッシングユニット２００の加算器ＡＤＤ１の一方入力には、この加算器ＡＤＤ１自身の出力が接続される。他方の入力には、データバス２１６Ｍを介して、プロセッシングユニット２０６の加算器ＡＤＤ１の出力が接続される。プロセッシングユニット２００の加算器ＡＤＤ０の一方の入力には、自分自身の出力が接続される。他方の入力には、データバス２１６Ｌを介して、プロセッシングユニット２０６の加算器ＡＤＤ０の出力が接続される。加算器ＡＤＤ０からのキャリーＣは加算器ＡＤＤ１のキャリー入力に与えられる。

【０１０６】プロセッシングユニット２０２においては、乗算器の２つの入力に、それぞれデータメモリからの２つのデータが与えられるようにセレクタが設定される。乗算器の２ｎビット出力のうち上位ｎビットはプロセッシングユニット２０２の加算器ＡＤＤ１の入力の一方に与えられる。加算器ＡＤＤ１の他方の入力には定数「０」が与えられる。プロセッシングユニット２０２の乗算器ＡＤＤ１の出力は、データバス２１２Ｍを介してプロセッシングユニット２０４の加算器ＡＤＤ１の入力の一方に接続される。プロセッシングユニット２０２の加算器ＡＤＤ０の入力の一方には、プロセッシングユニット２０２の乗算器の下位ｎビットが与えられる。他方の入力には、前述のとおり、プロセッシングユニット２００の乗算器の出力の上位ｎビットが与えられる。プロ

セッシングユニット２０２においても、加算器ＡＤＤ０のキャリーＣは加算器ＡＤＤ１に与えられる。

【０１０７】プロセッシングユニット２０４においては、乗算器の２つの入力にはデータメモリからの２つのデータが与えられる。乗算器の２ｎビットの出力のうち上位ｎビットは加算器ＡＤＤ１の入力の一方に接続され、下位ｎビットは加算器ＡＤＤ０の一方の入力に接続される。加算器ＡＤＤ１の他方の入力は、データバス２１２Ｍを介してプロセッシングユニット２０２の加算器ＡＤＤ１の出力に接続される。加算器ＡＤＤ０の他方の入力は、データバス２１２Ｌを介してプロセッシングユニット２０２の加算器ＡＤＤ０の出力に接続される。加算器ＡＤＤ０のキャリー出力Ｃは加算器ＡＤＤ１に与えられる。加算器ＡＤＤ１の出力は、データバス２１４Ｌを介してプロセッシングユニット２０６の加算器ＡＤＤ０の一方の入力に接続される。

【０１０８】プロセッシングユニット２０６においては、乗算器の２つの入力に、データメモリからの２つのデータが与えられるようにセレクタが設定される。乗算器の出力のうち上位ｎビットは加算器ＡＤＤ１の一方の入力に与えられる。下位ｎビットは加算器ＡＤＤ０の一方入力に接続される。加算器ＡＤＤ１の他方の入力には定数０が与えられる。加算器ＡＤＤ０の他方の入力には、前述のようにデータバス２１４Ｌを介してプロセッシングユニット２０４の加算器ＡＤＤ１の出力が接続される。加算器ＡＤＤ０の出力はデータバス２１６Ｌを介してプロセッシングユニット２００の加算器ＡＤＤ０の一方の入力に接続される。加算器ＡＤＤ１の出力はデータバス２１６Ｍを介してプロセッシングユニット２００の加算器ＡＤＤ１の一方の入力に接続される。加算器ＡＤＤ０のキャリー出力が、加算器ＡＤＤ１のキャリー入力に与えられる。

【０１０９】図８に示される接続において、演算は最下位のビット列の乗算から開始される。プロセッシングユニット２００の乗算器には、データメモリから上述のa_0およびb_0をそれぞれ与える。プロセッシングユニット２０２の乗算器には、データメモリを介して上述のa_1およびb_0をそれぞれ与える。プロセッシングユニット２０４の乗算器には、データメモリからa_0およびb_1をそれぞれ与える。プロセッシングユニット２０６の乗算器には、データメモリからa_1およびb_1をそれぞれ与える。

【０１１０】プロセッシングユニット２００における乗算結果の上位ｎビットが、データバス２１０Ｌを介してプロセッシングユニット２０２の加算器ＡＤＤ０に送られる。データバス２１０Ｌは、データバス２１０の下位ｎビットである。乗算結果の上位ｎビットをデータバスの下位ｎビットに出力するということは、実質的にデータをｎビット下位にシフトしたことと同等である。

【０１１１】プロセッシングユニット２０２の２つの加算器ＡＤＤ０、ＡＤＤ１では、乗算器の乗算結果$a_1 \times b_0$と、ｎビット下位にシフトされた$a_0 \times b_0$との間の２ｎビットの加算処理が行なわれることになる。加算結果の上位ｎビットはデータバス２１２Ｍを介してプロセッシングユニット２０４に、下位ｎビットはデータバス２１２Ｌを介してプロセッシングユニット２０４にそれぞれ与えられる。すなわちこの場合、データのシフトは行なわれない。

【０１１２】プロセッシングユニット２０４の乗算器の入力部分のセレクタは、データメモリからのデータa_0およびb_1を乗算器の入力に与えるように接続が設定される。乗算器の出力のうち上位ｎビットは加算器ＡＤＤ１に、下位ｎビットは加算器ＡＤＤ０にそれぞれ与えられる。プロセッシングユニット２０４の２つの加算器の間ではキャリーの入出力が行なわれるため、加算器ＡＤＤ０と加算器ＡＤＤ１とは、プロセッシングユニット２０２の出力する２ｎビットのデータに対して$a_0 \times b_1$を加算する２ｎビット加算処理を行なう。加算処理の上位ｎビットのみがデータバス２１４Ｌを介してプロセッシングユニット２０６に与えられる。

【０１１３】プロセッシングユニット２０６の乗算器の入力部分に設けられたセレクタは、データメモリからのデータa_1およびb_1を乗算器の２つの入力にそれぞれ与えるように接続が設定される。乗算器の出力の上位ｎビットは加算器ＡＤＤ１に与えられる。下位ｎビットは加算器ＡＤＤ０に与えられる。プロセッシングユニット２０４で行なわれた加算結果の上位ｎビットがデータバス２１４の下位ｎビットであるデータバス２１４Ｌを介して実質的にｎビット下方にシフトされてプロセッシングユニット２０６の加算器ＡＤＤ０に与えられる。したがってプロセッシングユニット２０６では、ｎビット下位にシフトされたプロセッシングユニット２０６の出力にさらに$a_1 \times b_1$を加算する２ｎビットの加算処理が行なわれる。

【０１１４】以上のようにして、２ｎビット同士の数ａ×ｂの乗算結果が、各クロックごとにプロセッシングユニット２０６の加算器出力に得られる。

【０１１５】さらに積和演算をする場合には、図８に示されるようにプロセッシングユニット２０６の加算器ＡＤＤ０およびＡＤＤ１の出力は、それぞれデータバス２１６Ｌおよび２１６Ｍを介してプロセッシングユニット２００の加算器ＡＤＤ０およびＡＤＤ１にそれぞれ与えられる。

【０１１６】この図８に示される接続例では、２ｎビット精度乗算を行なうのに、すべての乗算器とプロセッシングユニット２０２、２０４、２０６に含まれる加算器とが必要である。このとき同時に、プロセッシングユニット２００の加算器による２ｎビット精度演算も実行できる。したがってこの接続例ではこのプロセッサは２ｎビット精度の１回の乗算と２ｎビット精度の１回の加算

33

とを同時に実行可能である。図９に、この実施例のプロセッサにおいて、ＦＴＴ（高速フーリエ変換）に用いられるバタフライ演算を行なう場合の接続例を示す。演算はｎビット精度とする。バタフライ演算では、３つの複素数ａ、ｂ、ｃの間に、ｃ＋ａ×ｂとｃ－ａ×ｂで表わされる演算を行なう。ａｒ、ｂｒ、ｃｒをそれぞれａ、ｂ、ｃの実数部、ａｉ、ｂｉ、ｃｉを同じくａ、ｂ、ｃの虚数部、ｊを虚数単位とすると、ａ、ｂ、ｃはそれぞれ次のように表わされる。

【０１１７】ａ＝ａｒ＋ｊ・ａｉ
ｂ＝ｂｒ＋ｊ・ｂｉ
ｃ＝ｃｒ＋ｊ・ｃｉ
ｃ＋ａ×ｂとｃ－ａ×ｂとは、実数部および虚数部を合せて以下の４個の式により計算できる。

【０１１８】
ｃｒ＋（ａｒ×ｂｒ－ａｉ×ｂｉ）　…（１）
ｃｉ＋（ａｒ×ｂｉ＋ａｉ×ｂｒ）　…（２）
ｃｒ－（ａｒ×ｂｒ－ａｉ×ｂｉ）　…（３）
ｃｉ－（ａｒ×ｂｉ＋ａｉ×ｂｒ）　…（４）

この式（１）～（４）を求めるためには、見かけ上４回の演算を行なう必要があるが、これらには共通項が存在するので、実際に必要な演算は乗算４回と加算（減算）６回とである。このバタフライ演算を行なう接続は図９に示されるとおりである。

【０１１９】図９を参照して、プロセッシングユニット２００においては、乗算器の２つの入力にはデータメモリからの２つのデータがそれぞれ与えられる。プロセッシングユニット２００の加算器ＡＤＤ１の一方の入力には乗算器の出力の上位ｎビットが、他方の入力にはデータバス２２４Ｍがそれぞれ接続される。加算器ＡＤＤ１の出力はデータバス２２０Ｍに接続される。

【０１２０】プロセッシングユニット２０２の乗算器の２つの入力には、データメモリからの２つのデータが与えられる。プロセッシングユニット２０２の加算器ＡＤＤ１の入力の一方には乗算器の出力の上位ｎビットが、他方の入力にはデータバス２２６Ｍがそれぞれ接続される。加算器ＡＤＤ１の出力はデータバス２２２Ｍに接続される。

【０１２１】プロセッシングユニット２０４の乗算器の２つの入力には、データメモリからの２つのデータがそれぞれ与えられる。乗算器の出力の上位ｎビットはデータバス２２４Ｍを介してプロセッシングユニット２００に接続される。プロセッシングユニット２０４の加算器ＡＤＤ１の一方の入力は、データバス２２０Ｍを介してプロセッシングユニット２００に接続される。加算器ＡＤＤ１の他方の入力には、データメモリからのデータが与えられる。加算器ＡＤＤ０の入力も、加算器ＡＤＤ１の入力と共通に接続される。

【０１２２】プロセッシングユニット２０６の乗算器の２つの入力には、データメモリからの２つのデータがそ

34

れぞれ与えられる。乗算器の出力の上位ｎビットはデータバス２２６Ｍを介してプロセッシングユニット２０２に接続される。プロセッシングユニット２０６の２つの加算器ＡＤＤ０、ＡＤＤ１のそれぞれの一方の入力はデータバス２２２Ｍを介してプロセッシングユニット２０２に共通に接続される。それぞれの他方の入力には、データメモリからのデータが共通に与えられる。

【０１２３】プロセッシングユニット２０４および２０６の各々において、２個の加算器ＡＤＤ０およびＡＤＤ１の一方においては加算処理が、他方においては減算処理がそれぞれ行なわれる。

【０１２４】図９に示されるように接続されたプロセッサに、次のようにデータを与えることにより、プロセッシングユニット２０４および２０６の加算器の出力として、それぞれｃｒ＋（ａｒ・ｂｒ－ａｉ・ｂｉ）およびｃｒ－（ａｒ・ｂｒ－ａｉ・ｂｉ）と、ｃｉ＋（ａｒ・ｂｉ＋ａｉ・ｂｒ）およびｃｉ－（ａｒ・ｂｉ＋ａｉ・ｂｒ）が得られる。

【０１２５】プロセッシングユニット２００の乗算器の２つの入力には、データメモリからそれぞれａｒ、ｂｒを与える。プロセッシングユニット２０２の乗算器の２つの入力には、データメモリからそれぞれａｒ、ｂｉを与える。プロセッシングユニット２０４の乗算器の２つの入力には、データメモリからそれぞれａｉ、ｂｉを与える。プロセッシングユニット２０４の２つの加算器ＡＤＤ０、ＡＤＤ１の入力の一方には、データメモリからｃｒを与える。プロセッシングユニット２０６の乗算器の２つの入力には、データメモリからそれぞれデータａｉ、ｂｒを与える。プロセッシングユニット２０６の加算器ＡＤＤ０およびＡＤＤ１の入力の一方には、データメモリからｃｉを与える。

【０１２６】プロセッシングユニット２０４の乗算器からプロセッシングユニット２００へは、データバス２２４Ｍを介してａｉ・ｂｉが与えられる。プロセッシングユニット２００の加算器からプロセッシングユニット２０４へは、データバス２２０Ｍを介してａｒ・ｂｒ－ａｉ・ｂｉが与えられる。プロセッシングユニット２０４の２つの加算器では、ｃｒとａｒ・ｂｒ－ａｉ・ｂｉの加算および減算がそれぞれ行なわれる。したがって前述のとおり、加算器２つの出力にはそれぞれ、上述の式（１）および（３）が得られる。

【０１２７】プロセッシングユニット２０６の乗算器からは、データバス２２６Ｍを介してａｉ・ｂｒがプロセッシングユニット２０２に与えられる。プロセッシングユニット２０２の加算器ＡＤＤ１は、ａｒ・ｂｉ＋ａｉ・ｂｒを出力する。この出力はデータバス２２２Ｍを介してプロセッシングユニット２０６の２つの加算器に与えられる。プロセッシングユニット２０６の２つの加算器の一方ではｃｉとａｒ・ｂｉ＋ａｉ・ｂｒとの間の加算が、他方では減算が行なわれる。したがってプロセッ

シングユニット２０６の２つの加算器の出力として、上述の式（２）および（４）が得られる。

【０１２８】なお、この例においても、プロセッシングユニット２０６、２０４の乗算器の出力する２ｎビットデータは、加算器に入力される前に適当な丸め演算によりｎビットに丸められるものとする。

【０１２９】図１０に、ｎビット精度の積和演算を行なう場合の、このプロセッサ内のプロセッシングユニット間のデータバス接続を示す。ａｉ・ｂｉをｉを変化させながら加算する演算は、このプロセッサがプロセッシングユニットを４個含むために、4項単位で行なうことができる。まず、図１０に示されるプロセッサの接続例を説明する。

【０１３０】プロセッシングユニット２００の乗算器の２つの入力には、データメモリからの２つのデータが与えられる。プロセッシングユニット２００の乗算器の出力の上位ｎビットは加算器ＡＤＤ１の一方に入力に与えられる。加算器ＡＤＤ１のうちの他方の入力はデータバス２２４Ｍに接続される。加算器ＡＤＤ１の出力は、データバス２２０Ｍを介してプロセッシングユニット２０４に接続される。

【０１３１】プロセッシングユニット２０２の乗算器の２つの入力には、データメモリからの２つのデータがそれぞれ与えられる。乗算器の出力は加算器ＡＤＤ１の入力の一方に接続される。加算器ＡＤＤ１の他方の入力には、データバス２２６Ｍが接続される。加算器ＡＤＤ１の出力は、データバス２１２Ｍを介してプロセッシングユニット２０４に接続される。

【０１３２】プロセッシングユニット２０４の乗算器の２つの入力には、データメモリからの２つのデータが与えられる。乗算器の出力の上位ｎビットは、データバス２２４Ｍを介してプロセッシングユニット２００の加算器ＡＤＤ１の一方の入力に接続される。プロセッシングユニット２０４の加算器ＡＤＤ１の一方の入力はデータバス２１２Ｍに、他方の入力はデータバス２２０Ｍにそれぞれ接続される。加算器ＡＤＤ１の出力はデータバス２１４Ｍを介してプロセッシングユニット２０６に接続される。

【０１３３】プロセッシングユニット２０６の乗算器の２つの入力には、データメモリからの２つのデータが入力される。乗算器の出力の上位ｎビットはデータバス２２６Ｍを介してプロセッシングユニット２０２の加算器ＡＤＤ１の入力の一方に接続される。プロセッシングユニット２０６の加算器ＡＤＤ１の一方入力はデータバス２１４Ｍを介してプロセッシングユニット２０４に接続される。他方の入力は、加算器ＡＤＤ１自身の出力に接続される。

【０１３４】図１０に示されるように接続されたプロセッサでは、次のようにしてｎビット精度の積和演算が行なわれる。

【０１３５】プロセッシングユニット２００、２０２、２０４、２０６に、それぞれデータメモリから（a_0，b_0）、（a_1，b_1）、（a_2，b_2）、（a_3，b_3）を与える。プロセッシングユニット２００、２０２、２０４、２０６の乗算器の出力として$a_0 b_0$、$a_1 b_1$、$a_2 b_2$、$a_3 b_3$がそれぞれ得られる。

【０１３６】プロセッシングユニット２００の出力として$a_0 b_0 + a_2 b_2$が、プロセッシングユニット２０２の出力として$a_1 b_1 + a_3 b_3$がそれぞれ得られる。これらはプロセッシングユニット２０４の加算器ＡＤＤ１で加算され、$a_0 b_0 + a_1 b_1 + a_2 b_2 + a_3 b_3$が得られる。この例においても、各乗算器出力の２ｎビットは、加算器に入力される前に適当な丸め演算によりｎビットに丸められるものとする。

【０１３７】図１１は、この実施例のプロセッサにおいてｎビット精度の積和演算を別の方法により行なう場合の接続例を示す。図１０に示されるデータバス接続では、４項ごとの積和を得ていた。これに対し図１１に示される接続では、a_{i+1}、b_{i+1}は、a_i、b_iよりも１クロックずつ遅れて入力されるようにされている。その結果、各プロセッシングユニットの出力としては、１項ずつ加算した結果が得られる。最終結果は４項の積和ごとにプロセッシングユニット２００の加算器出力に得られるようになっている。

【０１３８】図１１に示される接続は次のようになっている。プロセッシングユニット２００においては、乗算器の２つの入力には、データメモリからの２つのデータ（a_0、b_0）が与えられる。乗算器出力の上位ｎビットはプロセッシングユニット２０２の加算器ＡＤＤ１の一方の入力に接続されている。プロセッシングユニット２００の加算器ＡＤＤ１の一方入力は、データバス２１６Ｍに接続されている。他方の入力は、加算器ＡＤＤ１自身の出力に接続されている。

【０１３９】プロセッシングユニット２０２において、乗算器の２つの入力には、データメモリからの２つのデータ（a_1、b_1）が与えられる。乗算器出力の上位ｎビットはプロセッシングユニット２０２の加算器ＡＤＤ１の残りの入力に接続されている。この加算器ＡＤＤ１の他方の入力は、前述のようにプロセッシングユニット２００の乗算器の出力の上位ｎビットに接続されている。加算器ＡＤＤ１の出力は、データバス２１２Ｍに接続されている。

【０１４０】プロセッシングユニット２０４において、乗算器の２つの入力には、データメモリからの２つのデータ（a_2、b_2）が与えられる。乗算器の出力の上位ｎビットは、プロセッシングユニット２０４の加算器ＡＤＤ１の一方の入力に接続される。加算器ＡＤＤ１の他方の入力は、データバス２１２Ｍを介してプロセッシングユニット２０２の加算器ＡＤＤ１の出力に接続されている。プロセッシングユニット２０４の加算器ＡＤＤ１

の出力は、データバス２１４Ｍに接続されている。

【０１４１】プロセッシングユニット２０６において、乗算器の２つの入力には、データメモリからの２つのデータ（a_3、b_3）が与えられる。乗算器出力の上位ｎビットはプロセッシングユニット２０６の加算器ＡＤＤ１の一方の入力に接続されている。加算器ＡＤＤ１の他方の入力は、データバス２１４Ｍを介してプロセッシングユニット２０４に接続されている。プロセッシングユニット２０６の加算器ＡＤＤ１の出力は、データバス２１６Ｍを介してプロセッシングユニット２００の加算器ＡＤＤ１の一方の入力に接続されている。

【０１４２】図１１に示されるように接続されたプロセッサでは、次のようにして積和演算が行なわれる。まず、プロセッシングユニット２００の乗算器の出力として$a_0 b_0$が得られる。次にプロセッシングユニット２０２の加算器の出力として、$a_0 b_0 + a_1 b_1$が得られる。次にプロセッシングユニット２０４の加算器の出力として、$a_0 b_0 + a_1 b_1 + a_2 b_2$が得られる。またプロセッシングユニット２０６の出力として、$a_0 b_0 + a_1 b_1 + a_2 b_2 + a_3 b_3$が得られる。

【０１４３】この例においても、各プロセッシングユニット内の乗算器出力の２ｎビットは、加算器に入力される前に適当な丸め演算によりｎビットに丸められるものとする。

【０１４４】以上のようにこの第１の実施例に係るプロセッサでは、各プロセッシングユニット内のセレクタを適切に切換えることにより、幅広い種類の演算を行なうことができる。各プロセッシングユニットの構造は全く同一であるため、プロセッサ内のレイアウトや、接続関係が簡明である。また各プロセッシングユニットの構造が同一であるために、これらプロセッシングユニットを制御するための制御命令に互換性があり、プロセッサの制御が容易になる。また各プロセッシングユニット間での２ｎビット幅のデータバスを用い、乗算結果の上位ｎビットを次の演算の下位ｎビットのデータとして他のプロセッシングユニットに与えることができる。データシフトのための手段を用いずに実質的にデータをｎビットシフトすることができ、簡略な回路でより多彩な演算を行なうことができる。シフト処理が不要なため処理も高速化される。

【０１４５】［第２の実施例］図１２に示されるのは、本発明の第２の実施例のプロセッサに用いられるプロセッシングユニットの１つ（ＰＵ００）である。このプロセッシングユニット５２０が図３に示される第１の実施例のプロセッシングユニット２００と異なるのは、セレクタ３５０からアドレスを受取り、セレクタ３６０の入力に該アドレスのデータを出力するためのＲＯＭ（読出専用メモリ）５３０を新たに含むことである。その他の点では、このプロセッシングユニット５２０は図３に示されるプロセッシングユニット２００と全く同一の構成である。したがってその他の部分についての詳しい説明はここでは繰り返さない。

【０１４６】プロセッサがこのような４個のプロセッシングユニットを含むものと仮定すると、他の３つのプロセッシングユニット（ＰＵ０１、ＰＵ１０、ＰＵ１１）も、このプロセッシングユニット５２０と全く同一の構成である。

【０１４７】図１２に示される第２の実施例のプロセッサのプロセッシングユニット５２０では、図３に示される第１の実施例のプロセッシングユニット２００の動作に加え、次のような演算処理を行なうことが可能となる。データメモリからＲＯＭ５３０のアドレスをこのプロセッシングユニット５２０に入力するものとする。セレクタ３５０によりそのアドレス信号を選択してＲＯＭ５３０に与える。ＲＯＭ５３０は、指定されたアドレスに格納されたデータをセレクタ３６０に与える。セレクタ３６０がこのデータをセレクトし乗算器３８０の一方の入力に与える。

【０１４８】ＲＯＭ５３０に格納するデータとしては、たとえばニュートン・ラプソン（Newton-Raphson）法による除算あるいは開平演算に用いられるデータが考えられる。たとえばニュートン・ラプソン法による除算においては、まず除数の逆数を乗算および加算による漸化式より求め、最後にその逆数に被除数を掛けることにより解を求める。この場合周知のように、漸化式により逆数を求める際の最初の近似値が十分近い値でなければ、漸化式の収束性は悪くなる。この近似値を予めＲＯＭに格納しておき、最初の近似値として演算に用いることで、漸化式の収束性が大きく向上し、上述した演算が効率よく行なえる。

【０１４９】［第３の実施例］本発明に係るプロセッサの第３の実施例の要部を図１３に示す。この第３の実施例のプロセッサは、第１の実施例とは異なる方法により命令を実行する。そのためにこの第３の実施例では、命令メモリとして第１の実施例に示される命令メモリ２６０など（図２参照）に代え、図１３および図１４に示される命令メモリ５４２を用いる。図１３および図１４においては、プロセッシングユニット２００を制御するための命令メモリ５４２のみを示したが、他のプロセッシングユニットを制御するための命令メモリもこの命令メモリ５４２と全く同一の構成である。

【０１５０】図１３を参照して、このプロセッサのプロセッシングユニット２００は、コントロール回路５４０により直接制御される。コントロール回路５４０は、命令メモリ５４２から与えられる演算器制御系命令に従いプロセッシングユニット２００を制御する。命令メモリ５４２はまた、データメモリ２７０にも接続されており、データメモリ制御系命令をコントロール回路５４０を介さずに直接データメモリ２７０に与えるためのものである。

39

【0151】図14を参照して、命令メモリ542は、複数の命令を格納するためのメモリ550と、メモリ550から読出される2つの命令を格納するための命令レジスタ552と、命令レジスタ552に接続された2つの入力を有する排他的OR（EXOR）回路556と、EXOR回路556の出力により制御され、命令レジスタ552に格納された2つの命令のうちの2番めのものをコントロール回路540（図13参照）に出力するか否かを選択するためのスイッチ554と、EXOR回路556の出力に接続され、メモリ550から次の読出すべき命令のアドレスを所定の論理に従って算出するためのアドレス演算論理558と、アドレス演算論理558の演算結果に従って、メモリ550内の連続する2つの読出アドレスを指定するためのプログラムカウンタ（PC）560とを含む。

【0152】図14においてメモリ550の左側に示される数字（100、101、102、103）は、各命令が格納されているアドレスを示す。メモリ550に格納された各命令572の先頭（第1ビット）570は、その命令がデータメモリ制御系命令であるか、演算器制御系命令であるかを示すフラグとなっている。図14に示される例の場合には第1ビット570が「0」であればメモリ制御系命令であり、「1」であれば演算器制御系命令であるものとする。

【0153】同様に命令レジスタ552も、命令582を格納する領域と、第1ビット580を格納する領域とを有している。命令レジスタ552は、こうした命令を格納する領域を2ヵ所有し、それぞれの第1ビットがEXOR回路556の2つの入力に接続されている。

【0154】図14に示される第3の実施例の命令メモリ542は、第1の実施例に示されるプロセッサの制御をより改善させるためのものである。第1の実施例のプロセッサでは、命令は命令メモリから1個ずつ取出され、それがデータメモリ制御命令か演算器制御系命令かが識別された後、その識別結果にしたがってデータメモリまたは各プロセッシングユニットの演算器の制御が行なわれていた。しかし、データメモリおよび演算器は相互に独立に動作（制御）することができる。したがって、データメモリ制御系命令と演算器制御系命令とが続いて命令メモリに格納されている場合には、この2つの命令を同時に実行した方が、順に実行するよりも効率がよい。この第3の実施例はこの点においてプロセッサの制御方法を改良したものである。

【0155】図14を参照して、プログラムカウンタ560のポインタ1（PC+A）が100番地を、ポインタ2（PC+A+1）が101番地をそれぞれ指定しているものとする。100番地と101番地の命令は命令レジスタ（IReg）552に同時に読込まれる。命令レジスタ552に格納された2つの命令の第1ビット580は、EXOR回路556の2つの入力にそれぞれ与

40

えられる。EXOR回路556の出力は、命令レジスタ552に格納された命令の2つの第1ビット580がともに「1」またはともに「0」であるときには「0」となり、そうでない場合には「1」となる。命令の先頭ビットを、データメモリ制御系命令では「0」に、演算器制御系命令では「1」にしておけば、このEXOR回路の出力により、2つの命令が同一系統かどうかを判断できる。

【0156】EXOR回路の出力が「1」であればスイッチ554は閉じられる。この場合には第2の命令（図14に示される例の場合にはPCNT命令）がコントロール回路540（図13参照）に送られる。また、EXOR回路556の出力が0のときにはスイッチ554は開き、第2の命令はコントロール回路540に送られない。一方、第1の命令は常にデータメモリに送られる。

【0157】EXOR回路556の出力が「1」の場合、アドレス演算論理558では、プログラムカウンタ560のAに2を、「0」であればAに1をそれぞれ代入し、プログラムカウンタ560に与える。すなわち、異なる系統の命令が読出された場合には、プログラムカウンタ560のポインタ1（PC+A）は、メモリ550の102番地を次に指定する。ポインタ2（PC+A+1）は103番地を指定する。したがって101番地の命令が改めて読出されることはない。

【0158】同一系統の命令が読出された場合には、ポインタ1（PC+A）は101番地を、ポインタ2（PC+A+1）は102番地を指定する。この場合には101番地および102番地の命令が同時に読出されて命令レジスタ552に格納され、上述した判断と判断結果に伴う命令の転送とアドレス演算とが行なわれる。

【0159】この第3の実施例では、命令は常に2個ずつ読出され、同一系統の命令であれば2番目の命令は実行されず、単にプログラムカウンタを1増加させてその2番目の命令を含む2つの命令を次に読出す。異なる系統の命令を読出した場合には一度に2つの命令を実行して、プログラムカウンタを2増加させ、読出された2番目の命令の次の2つの命令を次に読出すことになる。データメモリ制御系命令と演算器制御系命令とが連続して格納されている場合、これらを同時に実行することができ、プロセッサの動作効率が向上する。また前述のように各演算器は、直前に入力された演算器制御系命令に従って動作するので、データメモリの制御命令を繰り返し与えることにより、異なるデータに対する同一の演算を効率よく実行することができる。

【0160】［第4の実施例］この発明の第4の実施例に係るプロセッサを図15に示す。図15に示すプロセッサは、8個のプロセッシングユニット600、602、604、606、608、610、612、614と、8つのデータメモリ620、622、624、626、628、630、632、634とを含む。各プロ

41

セッシングユニットに付けられた数値は、２進数で表わされたそのプロセッシングユニットのアドレスを示す。データメモリ（ＤＭ）に付けられた数値も同様にそのアドレスを示す。

【０１６１】このプロセッサはさらに、データバス群６９０、６９２、６９４、６９６、６９８、７００、７０２、７０４、７０６により各データメモリ６２０、６２２、６２４、６２６、６２８、６３０、６３２、６３４に接続されたメモリ書込バス群６８０と、データメモリ６２０、６２２、６２４、６２６に接続され、かつデータバス群７１０、７１２、７１４、７１６によりそれぞれプロセッシングユニット６００、６０２、６０４、６０６に接続されたメモリ読出バス群６８２と、データメモリ６２８、６３０、６３２、６３４に接続され、かつデータバス群７１８、７２０、７２２、７２４によりそれぞれプロセッシングユニット６０８、６１０、６１２、６１４に接続されたメモリ読出バス群６８４とを含む。

【０１６２】さらにこのプロセッサでは、第１の実施例におけると同様に、アドレスが１ずつ異なるプロセッシングユニットを順次円環状に単方向に接続するためのデータバス６４０、６４２、６４４、６４６、６４８、６５０、６５２、６５４と、アドレスが２だけ異なるプロセッシングユニットを双方向に接続するためのデータバス群６６０、６６２、６６４、６６６、６６８、６７０、６７２、６７４とを含む。データバス群６６０、６６２、６６４、６６６、６６８、６７０、６７２、６７４は、それぞれ逆方向を向いた１対のデータバスを含んでいる。

【０１６３】図１５に示される構成では、各プロセッシングユニットにおいて、他のプロセッシングユニットとの間のデータバスとしては、３組の入力と３組の出力とがある。これは各プロセッシングユニット共通である。またプロセッシングユニットとデータメモリとはそれぞれ４個ずつの２つのグループに分類されている。第１のグループはプロセッシングユニット６００、６０２、６０４、６０６とデータメモリ６２０、６２２、６２４、６２６とを含み、第２のグループはプロセッシングユニット６０８、６１０、６１２、６１４と、データメモリ６２８、６３０、６３２、６３４とを含む。各グループのデータメモリとプロセッシングユニットとはそれぞれメモリ読出バス群６８２、６８４により接続されている。また各プロセッシングユニットとメモリ書込バス６８０とは、図示されていないメモリ書込バス群により接続され、各プロセッシングユニットから任意のデータメモリに対してデータを書込むことができる。このようにすることにより、グループ間の通信をデータメモリを介して行なうことができる。

【０１６４】図１５に示されるようにデータメモリをグループ化するのは、データメモリの読出バスをできるだけ短くするためである。データメモリの読出バスは長くなるとスピードが遅くなる。したがって、この実施例のようにプロセッシングユニット４個単位でグループ化してその長さを短くした方が動作速度上有利な場合が多い。また実際の応用における演算では、倍精度演算、バタフライ演算など、プロセッシングユニットを４項単位で使用する演算要求が多い。したがって図１５に示されるようにプロセッシングユニットとデータメモリとを４項単位でグループ化することにより、実際的で、かつ幅広い種類の演算に対応できるプロセッサを得ることができる。

42

【０１６５】図１５に示されるプロセッサでは、それぞれのグループにおいて倍精度の１乗算と１加算、バタフライ演算、４項ごとの積和演算がそれぞれ可能である。各プロセッシングユニットで用いる入力データはそのグループ内にあるので、グループ内でデータを共有できる。したがって図１５に示される構成と異なり、データメモリからのデータ読出バスをそのグループ内で閉じるようにしてもよい。

【０１６６】図１５に示されるプロセッサでは、各プロセッシングユニットの構造は相互に全く同一である。したがって、各プロセッシングユニットを制御する制御命令には互換性がある。またプロセッサを作製する上で、プロセッシングユニットやデータメモリのレイアウトが単純でよいという利点がある。また、第１の実施例と同様に、各プロセッシングユニット内のセレクタを、所望の演算を実現できるように切換えることにより、複数のプロセッシングユニットを用いた複雑な演算を、従来のものよりもより多種類実行することができる。

【０１６７】もちろん、いずれのプロセッシングユニットも任意のデータメモリからデータを読出せるようにデータメモリ読出バス群とプロセッシングユニットとを接続することも考えられる。たとえば８項ごとの積和演算は、図１５に示されるようにグループ化せず８つのプロセッシングユニットを１つのグループとして構成した方が効率はよい。しかしその場合には、データメモリからの読出速度が低下するおそれがある。また、４項ごとの演算を行なうような場合には、図１５に示されるようにグループ化した方が好ましい。

【０１６８】

【発明の効果】以上のように請求項１に記載の並列処理のためのプロセッシングユニットは制御手段の制御により種々の演算が可能なため、汎用的でありかつ制御手段の制御は比較的単純でよい。プロセッシングユニットに接続される入出力データバスの数がすべてのプロセッシングユニットで同一であり、また各プロセッシングユニットの構造が同一であるために、これらを複数個組合わせて並列処理プロセッサを作製するときのプロセッシングユニット相互の接続が容易であり、それらの制御に互換性を持たせることができる。

【０１６９】その結果、より幅広い種類の演算を簡単に制御で実現できる、並列処理のためのプロセッシングユニットを提供できる。

【０１７０】請求項２に記載のプロセッシングユニットは、ｎ×ｎビットの乗算器とｎ＋ｎビットの２つの加算器を用いて、大量のデータ処理が要求される演算における典型的な処理を行なうことができる。

【０１７１】その結果、より幅広い種類の大量の演算を簡単な制御で実現できる、並列処理のためのプロセッシングユニットを提供できる。

【０１７２】請求項３に記載のプロセッシングユニットでは、２ｎビットのデータ同士の演算を、それぞれ上位ｎビットと下位ｎビットとに分けてｎ×ｎビットの乗算器およびｎ＋ｎビットの加算器を用いて効率よく行なうことができる。

【０１７３】その結果、倍精度演算を含むより幅広い種類の演算を簡単な制御で実現できる、並列処理のためのプロセッシングユニットを提供できる。

【０１７４】請求項４に記載のプロセッシングユニットでは、２つの加算器の一方のキャリー出力を他方のキャリー入力とする場合と、しない場合とを選択できる。したがって、２ｎビットのデータ同士の加算と、ｎビットのデータ同士の２つの別個の加算とを効率よく実行することができる。

【０１７５】その結果、倍精度の加算処理を含むより幅広い種類の大量の演算を簡単な制御で実現できる、並列処理のためのプロセッシングユニットを提供できる。

【０１７６】請求項５に記載のプロセッシングユニットの加算器では、自己の加算結果を用いる演算、たとえば積和演算と、データバス経由のデータを用いた加算との双方を効率よく行なうことができる。

【０１７７】その結果、積和演算を含むより幅広い種類の大量の演算を簡単な制御で実現できる、並列処理のためのプロセッシングユニットを提供できる。

【０１７８】請求項６に記載のプロセッシングユニットの加算器は、乗算結果を加算の入力に用いる演算、たとえば積和演算と、データバス経由のデータを用いた加算との双方を効率よく行なうことができる。

【０１７９】その結果、積和演算を含むより幅広い種類の大量の演算を簡単な制御で実現できる、並列処理のためのプロセッシングユニットを提供できる。

【０１８０】請求項７に記載のプロセッシングユニットの加算器は、乗算結果や加算結果を加算の入力に用いる演算、たとえば積和演算と、データバス経由のデータを用いた加算との双方を効率よく行なうことができる。

【０１８１】その結果、積和演算を含むより幅広い種類の大量の演算を簡単な制御で実現できる、並列処理のためのプロセッシングユニットを提供できる。

【０１８２】請求項８に記載のプロセッシングユニットの乗算器では、記憶手段に格納されたデータを用いることにより処理の高速化を図れる処理、たとえばニュートン・ラプソン法による除算や、開平演算などを効率よく行なうための構成と、通常の演算を行なうための構成とを自由に選択できる。

【０１８３】その結果、ニュートン・ラプソン法や開平演算を含むより幅広い種類の大量の演算を簡単な制御で効率良く実現できる、並列処理のためのプロセッシングユニットを提供できる。

【０１８４】請求項９に記載の並列処理プロセッサでは、各プロセッシングユニットは、最低でも２以上の同じ数の入力と出力とを有する。入出力の数が多く、演算の自由度が高くなる。また、各プロセッシングユニットは同一の構造を有するため、プロセッサのレイアウトおよび制御の複雑さが低減される。

【０１８５】その結果、より幅広い種類の演算を簡単な制御で実現できる、簡単な構成の並列処理のためのプロセッサを提供できる。

【０１８６】請求項１０に記載の並列処理プロセッサの各プロセッシングユニットは、データ記憶手段からのデータと、他のプロセッシングユニットからのデータとのいずれにも、複数個の演算手段を用いた所定の処理を行ない、任意の演算手段の出力を他のプロセッシングユニットと、データ記憶手段との任意のものに出力できる。したがって、複数の同一構造のプロセッシングユニットを用いた多くの種類の演算を、互換性のある制御命令を用いて実行できる。

【０１８７】その結果、より幅広い種類の演算を簡単な制御で実現できる、並列処理のためのプロセッサを提供できる。

【０１８８】請求項１１に記載の並列処理プロセッサでは、データ記憶手段からプロセッシングユニットへの読出データバスは、プロセッシングユニットをグループ化せず全体に接続する場合と比較して短くすみ、プロセッシングユニットへのデータの機器が高速に行なえる。また１グループのプロセッシングユニットで共通のデータを用いた演算を実行できる。さらに各プロセッシングユニットからは任意のデータ記憶手段にデータを書込めるので、グループ間のデータ交換も可能で、複数のプロセッシングユニットを用いた多彩な演算を実行できる。また各プロセッシングユニットは同一構成で、互換性のある制御命令で制御することができる。

【０１８９】その結果、より幅広い種類の演算を簡単な制御で高速に実行できる、並列処理のためのプロセッサを提供できる。

【０１９０】請求項１２に記載の並列処理プロセッサの動作方法では、演算結果の上位ｎビットまたは下位ｎビットは、新たな演算のためのデータの下位ｎビットまたは上位ｎビットとして乗算手段および加算手段に与えることができ、多様な演算を行なうことができる。また、２つの加算手段の間でキャリーの入出力を行なうかどう

かを選択できるので、nビット精度の2つの加算処理と2nビット精度の1つの加算処理とを切換えて行なうことができる。したがって、非常に自由度の高い処理を行なうことができる。また各プロセッシングユニットの構成は同一で、かつ互換性のある制御命令で制御できる。

【0191】その結果、より幅広い種類の演算を簡単な制御で実行できる、並列処理のためのプロセッサの動作方法を提供できる。

【0192】請求項13に記載の並列処理プロセッサの動作方法では、あるプロセッシングユニット内の乗算手段または加算手段の出力の上位nビットまたは下位nビットが、他のプロセッシングユニット内の乗算手段または加算手段の下位nビットと上位nビットとの任意の一方に入力されるようにデータバスと乗算手段または加算手段との接続を設定できる。シフト手段を用いることなく多様な演算を行なうことが可能である。またシフト手段を用いる場合よりも動作が高速で、制御も単純である。

【0193】その結果、より幅広い種類の演算を簡単な制御で高速に実行できる、並列処理のためのプロセッサの動作方法を提供できる。

【0194】請求項14に記載の並列処理プロセッサの動作方法では、あるプロセッシングユニット内の乗算手段または加算手段の出力の上位nビットが、他のプロセッシングユニット内の乗算手段または加算手段の下位nビットに入力されるようにデータバスと乗算手段または加算手段との接続を設定できる。シフト手段を用いることなくデータを実質的にシフトでき、シフトのための時間も必要としない。シフト手段を用いることなく多様な演算を行なうことが可能である。またシフト手段を用いる場合よりも動作が高速で制御も単純である。

【0195】その結果、より幅広い種類の演算を簡単な制御で高速に実行できる、並列処理のためのプロセッサの動作方法を提供できる。

【0196】請求項15に記載の並列処理プロセッサでは、複雑な演算を各プロセッシングユニットごとに制御すればよく、並列処理プロセッサ全体の制御が容易になる。また、各プロセッシングユニットでは、制御手段により多様な演算処理を行なえ、かつ複数のプロセッシングユニットを組合わせることにより、プロセッサ全体としてさらに従来より幅広い処理を実現できる。

【0197】その結果、より幅広い種類の演算を簡単な制御で実現できる、並列処理のためのプロセッサを提供できる。

【0198】請求項16に記載の並列処理プロセッサでは、データ記憶手段から出力されるデータを変化させつつ一定の命令を繰り返し行なう場合などに、データ記憶手段を制御する命令のみを、その内容を変化させつつ繰り返し発行すれば、演算手段のための異なる制御命令をいくつも発行する必要がない。したがって、大量のデータに対する同一の演算を行なう場合の制御が容易である。また、制御命令を組合わせることで、各プロセッシングユニットにおいて多様な演算処理を行なうことができる。複数のプロセッシングユニットを組合わせることができ、さらにより複雑な演算を実現できる。

【0199】その結果、より幅広い種類の演算を簡単な制御で実現できる、並列処理のためのプロセッサを提供できる。

【0200】請求項17に記載の並列処理プロセッサでは、データ記憶手段の各々の2つの読出アドレスからの2つのデータを読出して所定の演算を行ない、結果を1つの書込アドレスにより指定された位置に書込できる。複数のプロセッシングユニットの間で、データ記憶手段を介してデータを授受しつつ、複雑な演算を実行できる。各プロセッシングユニットは同一構成であり、そのレイアウトは単純でよい。しかも互換性のある制御命令でプロセッシングユニットを制御でき、プロセッサの制御が簡略になる。

【0201】その結果、より幅広い種類の演算を簡単な制御で実現できる、並列処理のためのプロセッサを提供できる。

【0202】請求項18に記載の並列処理プロセッサでは、データ記憶手段から出力されるデータを変えつつ一定の命令を繰り返し行なう場合などに、データ記憶手段を制御する命令のみを、その内容を変化させつつ繰り返し発行すれば、演算手段のための同一の制御命令を繰り返し発行する必要がない。したがって大量のデータを処理する場合、プロセッサ全体の制御が単純になる。

【0203】その結果、より幅広い種類の大量の演算を簡単な制御で実現できる、並列処理のためのプロセッサを提供できる。

【0204】請求項19に記載の並列処理プロセッサでは、命令メモリから読出された2つの命令が同一の系統に属すれば、それらは同時に実行される。異なる系統であれば通常と同様の処理が行なわれる。一度に制御命令を1つしか読出さない場合と比較して、命令の実行速度は向上する。

【0205】その結果、より幅広い種類の演算を簡単な制御で高速に実行できる、並列処理のためのプロセッサを提供できる。

【図面の簡単な説明】

【図1】本発明の第1の実施例に係るプロセッサの構成を示す模式的ブロック図である。

【図2】第1の実施例のプロセッサのブロック図である。

【図3】第1の実施例のプロセッシングユニットのブロック図である。

【図4】第1の実施例におけるデータメモリとメモリ読出データバス群との間の接続を示すブロック図である。

【図5】制御命令の構成を示す模式図である。

47

【図６】命令メモリにおける命令の格納状態を示す模式図である。

【図７】本発明の第１の実施例のプロセッサによる第１の接続例を示す模式的ブロック図である。

【図８】本発明の第１の実施例のプロセッサによる第２の接続例を示す模式的ブロック図である。

【図９】本発明の第１の実施例のプロセッサによる第３の接続例を示す模式的ブロック図である。

【図１０】本発明の第１の実施例のプロセッサによる第４の接続例を示す模式的ブロック図である。

【図１１】本発明の第１の実施例のプロセッサによる第５の接続例を示す模式的ブロック図である。

【図１２】本発明の第２の実施例のプロセッサのプロセッシングユニットのブロック図である。

【図１３】本発明の第３の実施例のプロセッサの要部のブロック図である。

【図１４】本発明の第３の実施例における命令メモリの模式的ブロック図である。

【図１５】本発明の第４の実施例のプロセッサの構成を示す模式的ブロック図である。

【図１６】従来の並列処理プロセッサを示す模式的ブロック図である。

【図１７】図１６に示す従来のプロセッサの各プロセッシングユニットの構成を示すブロック図である。

48

【図１８】図１６および図１７に示される従来のプロセッサのプロセッシングユニットの接続例を示す模式的ブロック図である。

【図１９】従来のプロセッサのプロセッシングユニット相互の間の接続例を示す模式的ブロック図である。

【符号の説明】

２００、２０２、２０４、２０６　プロセッシングユニット

２１０、２１２、２１４、２１６　単方向入力データバス

２２０、２２２、２２４、２２６　単方向入力データバス

２５０、２５２、２５４、２５６　コントロール回路

２６０、２６２、２６４、２６６　命令メモリ

２８０　メモリ読出データバス群

２８２　メモリ書込データバス群

２７０、２７２、２７４、２７６　データメモリ

３５０、３５２、３５４、３５６　セレクタ

３６０、３６２、３６４、３６６、３６８、３７０　セレクタ

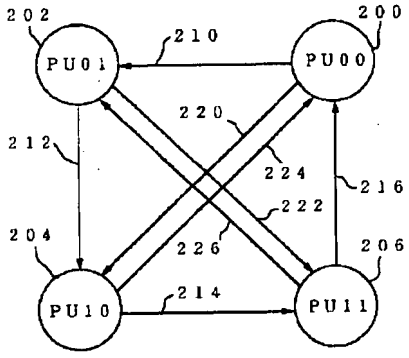
３８０　乗算器

３８２、３８４　加算器

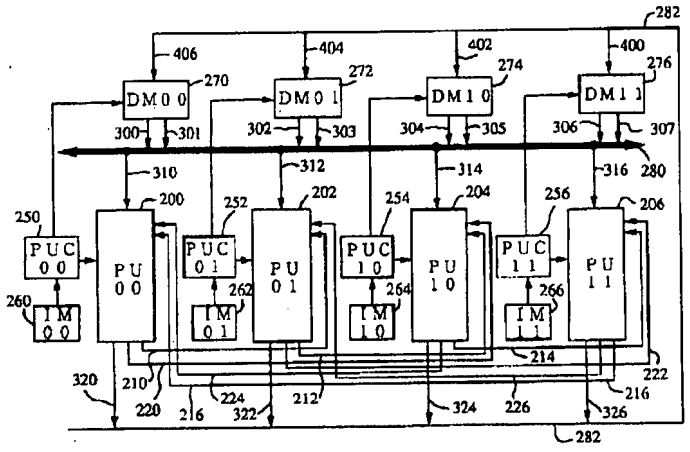
３９０　セレクタ

３９２　クロスバースイッチ

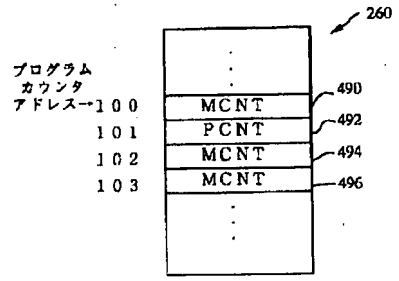
【図１】



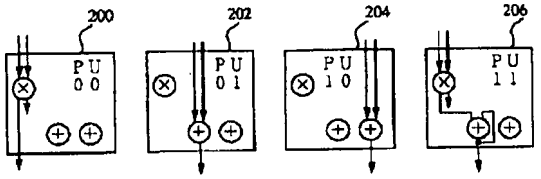
【図２】



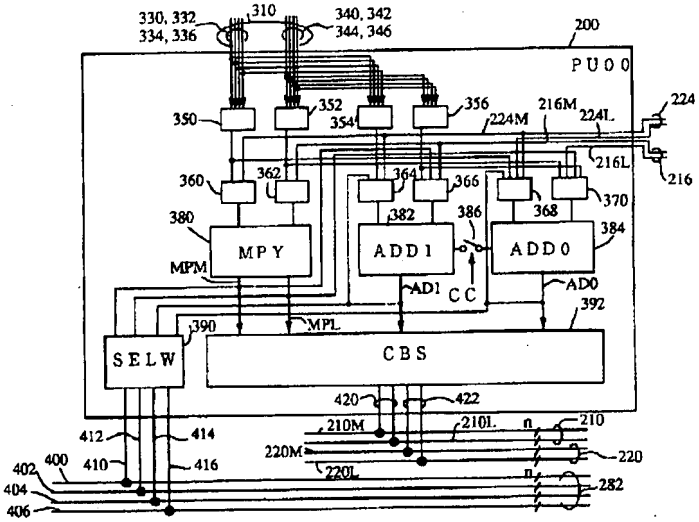
【図６】



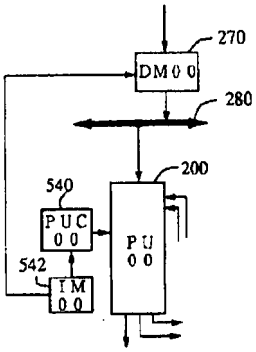
【図７】



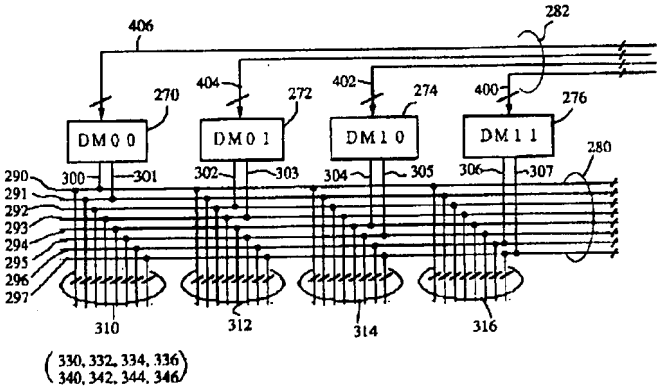
【図３】



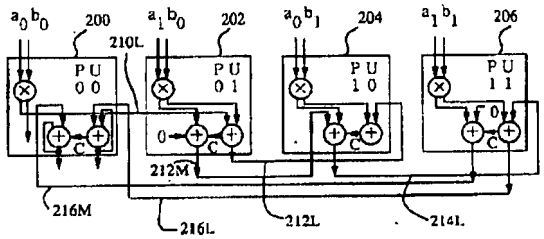
【図１３】



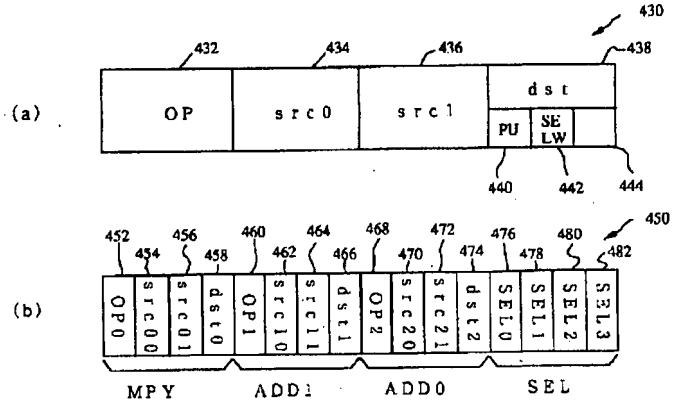
【図４】



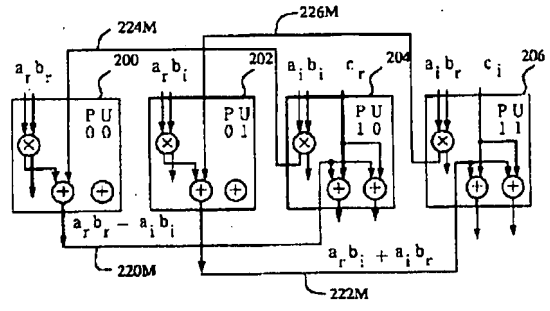
【図８】



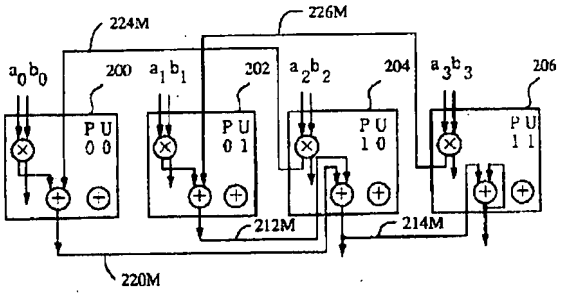
【図５】



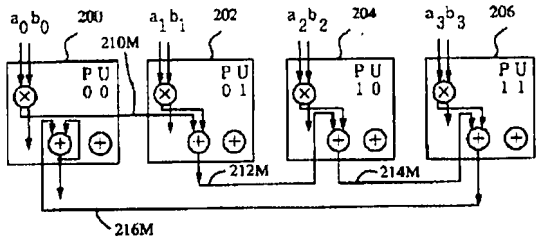
【図９】



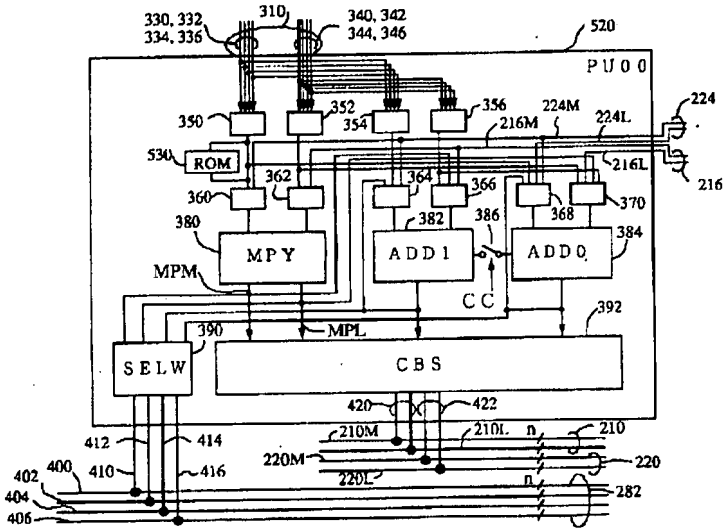
【図１０】



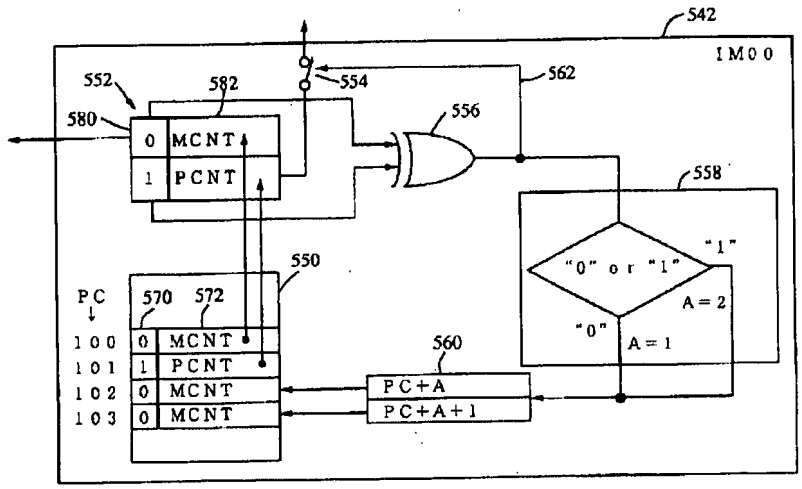
【図１１】



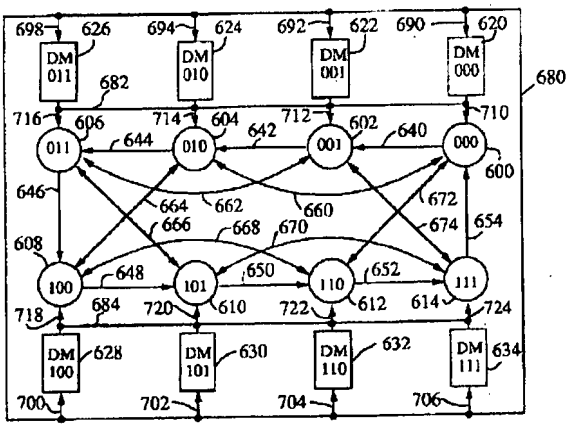
【図１２】



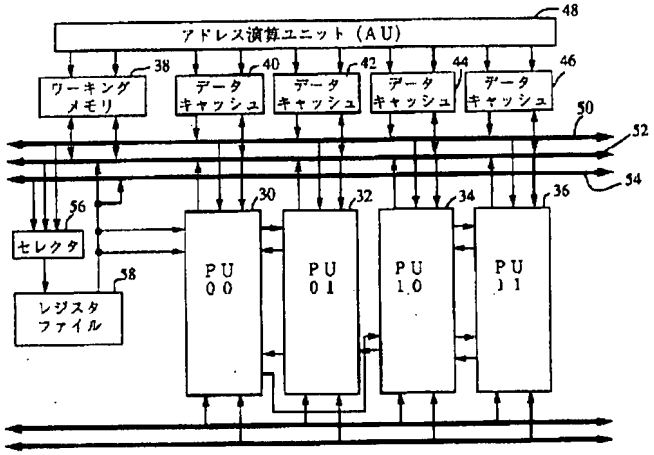
【図１４】



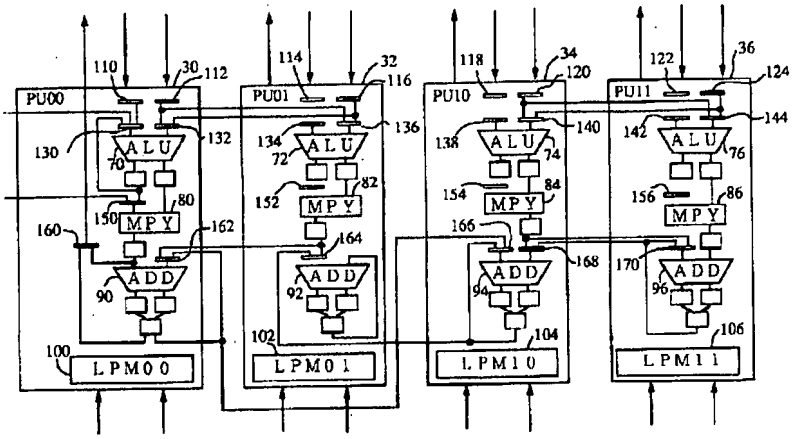
【図１５】



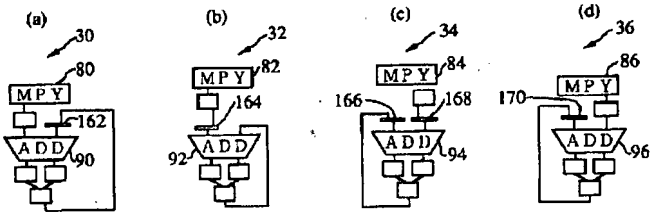
【図１６】



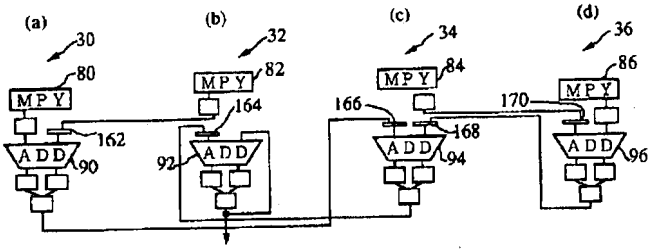
【図１７】



【図１８】



【図１９】


(a)
30
80
MPY
162
ADD
90
(b)
32
MPY
82
164
92
ADD
(c)
34
MPY
84
166
168
ADD
94
(d)
36
86
MPY
170
ADD
96

JAPANESE PATENT OFFICE
PATENT JOURNAL (A)
KOKAI PATENT APPLICATION NO. HEI 7[1995]-64789

Int. Cl.[6]:	G 06 F 9/38
Filing No.:	Hei 5[1993]-210783
Filing Date:	August 25, 1993
Publication Date:	March 10, 1995
No. of Claims:	19 (Total of 29 pages; OL)
Examination Request:	Not filed

PARALLEL PROCESSING PROCESSOR, ITS PROCESSING UNIT, AND OPERATION METHOD FOR SAID PARALLEL PROCESSING PROCESSOR

Inventor:	Yasunobu Nakase LSI Research Lab., Mitsubishi Electric Corp. 4-1 Mizuhara, Itami-shi, Hyogo-ken
Applicant:	000006013 Mitsubishi Electric Corp. 2-2-3 Marunouchi, Chiyoda-ku, Tokyo
Agents:	Kuro Fukami, patent attorney, and 3 others

[There are no amendments to this patent.]

Abstract

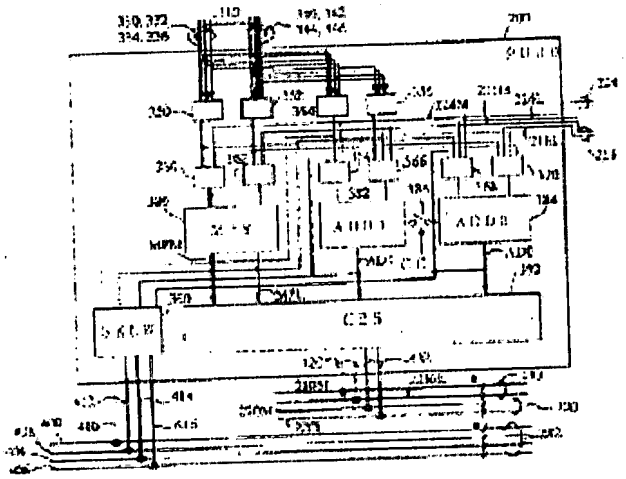
Purpose

To provide a general purpose parallel processing processor that can be controlled easily.

Constitution

Each of the plural processing units of the processor includes computing elements 380, 382, 384, selectors 360-370 that provide the high-order n bits or low-order n bits of input data

buses 216, 224 from another unit, and a crossbar switch 392 that outputs the outputs of computing elements 380, 382, 384 to any one of data buses 210, 220 to another unit. Various arithmetic operations can be carried out by setting the connection of each selector in advance. Also, various arithmetic operations can be realized for the entire processor by data exchange via a data bus with another processing unit. The constitution of each processing unit is the same. The layout is easy, and the processing units can be controlled by interchangeable control instructions.



Claims

1. A processing unit used for parallel processing having the following: plural operation means, which have plural inputs, that respectively carry out a prescribed arithmetic operation between the provided data, and output the result;

an input data bus selecting means, which is connected to plural monodirectional input data buses, that selects one of the aforementioned plural monodirectional input data buses in a controllable manner for each input of the aforementioned plural operation means, and provides part of the data provided via the selected monodirectional input data bus to the aforementioned input;

an output data bus selecting means, which has inputs connected to the outputs of the aforementioned plural operation means and outputs connected to the same number of monodirectional output data buses as the aforementioned monodirectional input data buses and outputs each output of the aforementioned operation means to one of the aforementioned monodirectional output data buses; and

a control means, which is used to control the path of the data depending on the aforementioned input data bus selecting means and the output data bus selecting means in order

to realize desired composite arithmetic operation by using the aforementioned plural operation means.

2. The processing unit used for parallel processing described in Claim 1, characterized by the fact that the aforementioned plural operation means have two n-bit inputs and include a multiplier, which multiplies two provided data and outputs a 2n-bit result, and two adders, which add the two provided data and output the n-bit result.

3. The processing unit used for parallel processing described in Claim 2, characterized by the following facts: each of the aforementioned plural monodirectional input data buses and monodirectional output data buses has a 2n-bit width;

the aforementioned input data bus selecting means includes a means which selects one of the aforementioned plural monodirectional input data buses in a controllable manner for each input of the aforementioned plural operation means and provides the high-order n bits or low-order n bits of the data provided via the selected monodirectional input data bus to the aforementioned input;

the aforementioned output data bus selecting means includes a means which outputs each output of the aforementioned operation means to high-order n bits and/or low-order n bits of the aforementioned monodirectional output data bus.

4. The processing unit used for parallel processing described in Claim 2, characterized by the fact that one of the aforementioned two adders has a carry output,

the other one of the aforementioned two adders has a carry input, and

the processing unit has a means used for intermitting the aforementioned carry output and carry input in a controllable manner.

5. The processing unit used for parallel processing described in Claim 2, characterized by the fact that the aforementioned input data bus selecting means includes a means which, for at least one input of one of the aforementioned two adders, selects one of the aforementioned plural monodirectional input data buses or the output of that adder itself in a controllable manner and provides the data provdied via the selected monodirectional input data bus or a part of the output of that adder itself to the aforementioned input.

6. The processing unit used for parallel processing described in Claim 2, characterized by the fact that the aforementioned input data bus selecting means includes a means which selects one of the aforementioned plural monodirectional input data buses or the output of the aforementioned multiplier for at least one input of one of the aforementioned two adders in a controllable manner and provides the data provided via the selected monodirectional input data bus or a part of the output of the aforementioned multiplier to the aforementioned input.

7. The processing unit used for parallel processing described in Claim 2, characterized by the fact that the aforementioned input data bus selecting means includes a means which, for at

least one input of one of the aforementioned two adders, selects one of the aforementioned plural monodirectional input data buses or the output of that adder itself or a part of the output of the aforementioned multiplier in a controllable manner and provides the data provided via the selected monodirectional input data bus or the output of that adder itself or a part of the output of the aforementioned multiplier to the input.

8. The processing unit used for parallel processing described in Claim 2, characterized by the following facts: the processing unit also includes a read only memory means used for storing prescribed information in advance;

the aforementioned input data bus selecting means includes a means which selects one of the aforementioned plural monodirectional input data buses or the output of the aforementioned read only memory means for at least one input of the aforementioned multiplier and provides the data provided via the selected monodirectional input data bus or part of the output of the aforementioned memory means to the aforementioned input.

9. A parallel processing processor characterized by the following facts: the parallel processing processor has

n processing units;

first monodirectional data buses used to connect the adjacent processing units sequentially in a loop in a prescribed direction;

second monodirectional data buses used to connect every other adjacent processing units sequentially bidirectionally in a loop;

said n is a multiple of 4;

each of the aforementioned processing units includes the following:

plural operation means, which have plural inputs, that respectively carry out a prescribed arithmetic operation between the provided data, and output the result;

an input data bus selecting means, which is connected to the first monodirectional data bus that provides input from the adjacent processing unit and to the input data bus to the processing unit concerned out of the aforementioned second monodirectional data buses, that selects one of said first and second monodirectional data buses for each input of the aforementioned plural operation means, and provides part of the data provided via the selected monodirectional data bus to the aforementioned input;

an output data bus selecting means, which has an input connected to the outputs of the aforementioned plural operation means and an output connected to the aforementioned first monodirectional data bus, that sends output to the adjacent processing unit and to the output data bus from the processing unit concerned out of the aforementioned second monodirectional data buses and outputs each output of the aforementioned operation means to one of the aforementioned first and second monodirectional data buses; and

a control means, which is used to control the path of the data depending on the aforementioned input data bus selecting means and the output data bus selecting means in order to realize desired composite arithmetic operation by using the aforementioned plural operation means.

10. The parallel processing processor described in Claim 9, characterized by the following facts: the parallel processing processor also includes

data storage means in the same number of the aforementioned processing units, each of which can output two data at a time, as well as

plural read data buses and write data buses used for connecting each of the aforementioned data storage means to each of the aforementioned processing units;

in each processing unit,

the aforementioned input data bus selecting means includes a means, which is connected to the first monodirectional data bus that provides input from the adjacent processing unit, to the input data bus to the processing unit concerned out of the aforementioned second monodirectional data buses, and to the aforementioned plural read data buses, that selects one data bus from said first and second monodirectional data buses and said read data buses for each input of the aforementioned plural operation means, and provides part of the data provided via the selected data bus to the aforementioned input;

the aforementioned output data bus selecting means includes a means, which has an input connected to the outputs of the aforementioned plural operation means and an output connected to the aforementioned first monodirectional data bus that sends output to the adjacent processing unit, to the output data bus from the processing unit concerned out of the aforementioned second monodirectional data buses, and to the aforementioned write data buses and outputs each output of the aforementioned operation means to one of the aforementioned data buses.

11. The parallel processing processor described in Claim 9, characterized by the following facts: the parallel processing processor also includes

data storage means in the same number of the aforementioned processing units, each of which can output two data at a time, as well as

plural read data buses and write data buses used for connecting each of the aforementioned data storage means to each of the aforementioned processing units;

the aforementioned plural processing units are divided into plural groups, each of which includes processing units in a number as a multiple of 4;

one-to-one correspondence is established between the aforementioned plural processing units and the aforementioned plural data storage means;

in each processing unit,

the aforementioned input data bus selecting means includes a means, which is connected to the first monodirectional data bus that provides input from the adjacent processing unit, to the input data bus to the processing unit concerned out of the aforementioned second monodirectional data buses, and to read data bus from the data storage means corresponding to the processing unit concerned in the group including that processing unit among the aforementioned plural read data buses, that selects one data bus from said first and second monodirectional data buses and said read data buses for each input of the aforementioned plural operation means, and provides part of the data provided via the selected data bus to the aforementioned input;

the aforementioned output data bus selecting means includes a means, which has inputs connected to the outputs of the aforementioned plural operation means and an output connected to the aforementioned first monodirectional data bus that sends output to the adjacent processing unit, to the output data bus from the processing unit concerned out of the aforementioned second monodirectional data buses, and to all of the aforementioned write data buses and outputs each output of the aforementioned operation means to one of the aforementioned data buses.

12. An operation method characterized by the following facts: the operation method is used to carry out desired arithmetic operation in a parallel processing processor including the following:

four processing units;

four first monodirectional data buses used to connect the adjacent processing units sequentially in a loop in a prescribed direction; and

four second monodirectional data buses used to connect every other adjacent processing units bidirectionally;

each processing unit includes the following:

a multiplier, which has 2 n-bit inputs and multiplies the provided data and outputs a 2n-bit result;

a first adder and a second adder, each of which has 2 n-bit inputs and adds the provided data and outputs an n-bit result;

a carry switching means used to provide the carry output of the first adder to the carry input of the second adder in a controllable manner;

an input data bus selecting means, which is connected to the aforementioned first monodirectional data bus that provides input from the adjacent processing unit and to the input data bus to the processing unit concerned from the aforementioned second monodirectional data buses, that selects the first or second monodirectional data bus for each input of the aforementioned plural operation means, and provides part of the data provided via the selected

monodirectional data bus to the aforementioned inputs of the aforementioned multiplier and adders;

an output data bus selecting means, which has an input connected to the output of the aforementioned multiplier and the outputs of the aforementioned adders, to the aforementioned first monodirectional data bus that provides output to the adjacent processing unit, and to the output data bus from the processing unit concerned out of the aforementioned second monodirectional data buses and outputs each of the outputs of the aforementioned multiplier and adders to one of the first and second monodirectional data buses; and

a control means, which controls the path of the data depending on the aforementioned input data bus selecting means and the output data bus selecting means in order to realize desired composite arithmetic operation by using the aforementioned multiplier and adders;

the operation method has the following steps:

the data needed for the operation are provided to the aforementioned input data bus selecting means;

the aforementioned data are divided for every high-order n bit and low-order n bit by the aforementioned input data bus selecting means and are provided to any two inputs of the aforementioned multiplier and adders of the aforementioned four processing units;

the aforementioned carry switching means of all of the processing units are set corresponding to the aforementioned desired arithmetic operation;

the aforementioned output data bus selecting means and input data bus selecting means of each of the processing units are controlled to set the connection between the multiplier and the first and second adders of each processing unit in order to realize the aforementioned desired arithmetic operation;

the aforementioned output data bus selecting means is controlled such that the prescribed part of the prescribed output among the outputs of the aforementioned multiplier and adders of the prescribed processing unit determined by the aforementioned desired arithmetic operation is output to the desired data bus so that the result of the arithmetic operation can be output to the desired data bus.

13. The operation method described in Claim 12, characterized by the fact that the step of setting the connection includes a step in which the data path depending on the output data bus selecting means and input data bus selecting means of the processing units is set such that the high-order n bits in the output of the multiplier or adders in a certain processing unit are input to the low-order n bits of the multiplier or adders in another processing unit.

14. The operation method described in Claim 12, characterized by the fact that the step of setting the connection includes a step in which the data path depending on the output data bus selecting means and input data bus selecting means of the processing units is set such that the

high-order n bits in the output of the multiplier or adders in a certain processing unit are output to the low-order n bits of the multiplier or adders in another processing unit.

15. The parallel processing processor described in Claim 9, characterized by also including plural instruction storage means, which are prepared for each processing unit and are used to store control instructions carried out by the aforementioned control means.

16. The parallel processing processor described in Claim 15, characterized by the fact that the aforementioned control instructions are classified into two systems, that is, a first type of control instructions used for controlling the corresponding data storage means and a second type of control instructions used for controlling the aforementioned operation means by the aforementioned control means.

17. The parallel processing processor described in Claim 16, characterized by the fact that the first type of control instruction specifies the two reading addresses and one writing address of each data storage means.

18. The parallel processing processor described in Claim 16, characterized by the fact that the aforementioned control means, the aforementioned operation means, input data bus selecting means, and output data bus selecting means are used according to the previously provided second type of control instruction until the second type of instruction is changed.

19. The parallel processing processor described in Claim 16, characterized by the fact that the aforementioned instruction storage means includes

an instruction memory used to store plural instructions,

a program counter used to specify the reading address of the instruction memory,

a means used to read two instructions at a time with the address specified by the aforementioned program counter as the header,

a means used to determine whether the two read instructions belong to the same system,

a means used to provide the control instructions to the aforementioned control means and/or data storage means based on the judgment result, and

a means used to increment the counted value of the aforementioned program counter by 1 or 2 based on the judgment result.

Detailed explanation of the invention

[0001]

Industrial application field

The present invention pertains to the improvement of a parallel processing processor constituted with plural processing units. In particular, the present invention pertains to a parallel processing processor which can be controlled easily to handle a wide range of arithmetic operations, its processing unit, and the operation method of the parallel processing processor.

[0002]

Prior art

The configuration of a conventional parallel processing processor (simply referred to as "processor" hereinafter) is shown in Figure 16 and 17. This processor is formed by extracting and simplifying the main operation parts related to the invention of this patent application among the block configurations of processor published in "ISSCC Digest of Technical Papers," (February 1991, p. 252-253).

[0003]

As shown in Figure 16, the processor includes 4 processing units PU00-PU11 (represented by symbols, 30, 32, 34, 36 in Figure 16), address computing unit (AU) 48, working memory 38, data cache memories 40, 42, 44, 46. The access to data cache memories 40, 42, 44, 46 and working memory 38 from each processing unit is carried out via a total of 10 buses, that is, 4 cache memory read only buses 50 and 6 read/write buses 42. This processor also includes register file 58 and selector 56. Selector 56 and register file 58 are used to perform data exchange between the processing units. The data read from cache memory read only buses 50, read/write buses 52, and SBUS 54 can be output to each of processing units 30, 32, 34, 36 and buses 52, 54 via register file 58.

[0004]

As shown in Figure 17, processing units 30, 32, 34, 36 have similar configuration or configurations that are slightly different from each other. Processing units 30, 32, 34, 36 include computing elements ALU 70, 72, 74, 76, multipliers MPY 80, 82, 84, 86, and adders ADD 90, 92, 94, 96. Said processing units 30, 32, 34, 36 include selectors used for selecting the inputs to the ALU, multipliers, and adders. This will be described later. In Figure 17, however, the inputs/outputs of the selectors are shown in a simplified way in the figure.

[0005]

As shown in Figure 17, for example, processing unit 30 includes 5:1 selectors 110 and 112. Processing unit 30 also includes selector 130, which is used to select one output from one of the outputs of ALU 70, the output from register file 58, and the output of selector 110 and provides it to one of the inputs of ALU 70, and selector 132, which is used to select the output of selector 112 or one of the data provided from processing unit 32 described later and provides it to the other input of ALU 70. Processing unit 30 also includes selector 150, which is used to select one of the outputs of ALU 70 or the output of register file 58 and provides it to one of the

inputs of MPY 80, and selector 162, which selects one of the outputs of ADD 90 and provides it to one of the inputs of ADD 90. Processing unit 30 also includes selector 160. Said selector 160 is used to select the output of MPY 80 or the output of ADD 90 and outputs it to data bus 52.

[0006]

Similarly, processing unit 32 includes selectors 114, 116, 134, 136, 152, 164. Processing unit 134 includes selectors 118, 120, 138, 140, 154, 166, 168. Processing units 32, 34, 36 also include the same selectors as selector 160. They, however, are not shown in Figure 17, in order to simplify the figure.

[0007]

Selectors 114, 118, 122 have the same function as selector 110. Selectors 116, 120, 124 have the same function as selector 112. Selectors 134, 138, 142 have the same function as selector 130. Selector 136 selects the output of selector 116 or the output of selector 112 and provides it to ALU 72. Selector 144 selects the output of selector 124 or the output of selector 120 and provides it to ALU 76. Selectors 152, 154, 156 have the same function as selector 150. Selector 164 selects the output of processing unit 34 or the output of multiplier 82 and provides it to adder 92. Selector 166 selects the output of processing unit 30 or the output of adder 94 and provides it to adder 94. Selector 168 selects the output of multiplier 84 or the output of processing unit 36 and provides it to adder 94. Selector 170 selects the output of multiplier 84 or the output of adder 96 and provides it to adder 96.

[0008]

Local instruction memories LPM00, 01, 10, 11 (represented by symbols 100, 102, 104, 106 in Figure 17) used for controlling the processing units are arranged in processing units 30, 32, 34, 36.

[0009]

Address computing unit 48 is used to compute the read and write addresses for memories 38, 40, 42, 44, 46.

[0010]

As shown in Figures 16 and 17, in the conventional processor, the configurations of the processing units are different from each other. The connection between the processing units also has a special form based on the arithmetic operation as the processing object.

[0011]

In the following, the operation of the conventional processor will be explained based on Figures 18 and 19. The bus connection between the processing units can be selected from two kinds of configurations shown in Figures 18 and 19. In the example shown in Figure 18, the various selectors are set such that there is no data bus connection between processing units 30, 32, 34, 36. A sum of products operation is performed by each processing unit.

[0012]

In the example shown in Figure 19, the output of the multiplier 82 in processing unit 32 is provided to the input of the adder 90 in processing unit 30. The output of adder 90 is provided to the input of the adder 94 in processing unit 34. On the other hand, the output of the multiplier 84 in processing unit 34 is provided to one of the inputs of the adder 96 in processing unit 36. The output of adder 96 is provided to the other input of the adder 194 in processing unit 34. The output of adder 94 is provided to one of the inputs of the adder 92 in processing unit 32. With the connection shown in Figure 19, the sum of products can be computed every 4 items. The result of the sum of products of every four items is obtained as the output of processing unit 32.

[0013]

The control of the processor is performed in a unit of a total of five instructions, that is, one setup instruction and four instructions used for controlling the processing units. Each instruction has 32 bits. Five instructions have a total of 160 bits.

[0014]

The setup instruction controls 5:1 selectors 110, 112, 114, 116, 118, 120, 122, 124 in the input parts of the processing units or the selectors used for setting the connection of data buses between the processing units. The processing unit control instruction generates the addresses of memories 38, 40, 42, 44, 46 or designate the addresses of local instruction memories 100, 102, 104, 106. The local instructions included in local instruction memories 100, 102, 104, 106 are used to specify the content of the arithmetic operation performed by the computing elements.

[0015]

Problems to be solved by the invention

The conventional processor shown in Figures 16-19 has the following problem. This processor was developed in order to increase the efficiency of sum-of-products operations for compressing dynamic images. Therefore, it is difficult to apply this processor to the fields other than dynamic image compression even for the processing requiring a large amount of

computation. Besides sum-of-products operations, other arithmetic operations requiring large amount of data processing include butterfly operations used for FFT (high-speed Fourier transform) and double precision operations used for scientific and technical calculations. For double precision multiplication (2n bits), 4 nxn multipliers and 3 2n+2n-bits adders are needed. The processor shown in Figures 16-19, however, can only be used to carry out processing for compression of dynamic images. Even its hardware is realized with a fixed algorithm. This is because the control becomes complicated when the aforementioned various operations are carried out in a processor having plural processing units. Consequently, very few of the conventional processors are general purpose processors. Also, the control and the hardware are complicated.

[0016]

The purpose of the present invention is to solve the aforementioned problem by providing a parallel processing processor which can carry out parallel processing efficiently by using plural processing units and can perform a wide range of operations with a relatively simple control method, and its processing unit and the operation method of the processor.

[0017]

Means to solve the problems

The processing unit used for parallel processing described in Claim 1 includes the following: plural operation means, which have plural inputs, that respectively carry out a prescribed arithmetic operation between the provided data, and output the result; an input data bus selecting means, which is connected to plural monodirectional input data buses, that selects one of the aforementioned plural monodirectional input data buses in a controllable manner for each input of the aforementioned plural operation means, and provides part of the data provided via the selected monodirectional input data bus to the aforementioned input; an output data bus selecting means, which has inputs connected to the outputs of the aforementioned plural operation means and outputs connected to the same number of monodirectional output data buses as the aforementioned monodirectioanl input data buses and outputs each output of the aforementioned operation means to one of the aforementioned monodirectional output data buses; and a control means, which is used to control the path of the data depending on the aforementioned input data bus selecting means and the output data bus selecting means in order to realize desired composite arithmetic operation by using the aforementioned plural operation means.

[0018]

The processing unit described in Claim 2 is based on Claim 1 and is characterized by the fact that the aforementioned plural operation means have two n-bit inputs and include a multiplier, which multiplies two provided data and outputs a 2n-bit result, and two adders, which add the two provided data and output the n-bit result.

[0019]

The processing unit described in Claim 3 is based on Claim 2 and is characterized by the following facts: each of the aforementioned plural monodirectional input data buses and monodirectional output data buses has a 2n-bit width; the aforementioned input data bus selecting means includes a means which selects one of the aforementioned plural monodirectional input data buses in a controllable manner for each input of the aforementioned plural operation means and provides the high-order n bits or low-order n bits of the data provided via the selected monodirectional input data bus to the aforementioned input; the aforementioned output data bus selecting means includes a means which outputs each output of the aforementioned operation means to high-order n bits and/or low-order n bits of the aforementioned monodirectional output data bus.

[0020]

The processing unit described in Claim 4 is based on Claim 2 and is characterized by the fact that one of the aforementioned two adders has a carry output, the other one of the aforementioned two adders has a carry input, and the processing unit has a means used for intermitting the aforementioned carry output and carry input in a controllable manner.

[0021]

The processing unit described in Claim 5 is based on Claim 2 and is characterized by the fact that the aforementioned input data bus selecting means includes a means which, for at least one input of one of the aforementioned two adders, selects one of the aforementioned plural monodirectional input data buses or the output of that adder itself in a controllable manner and provides the data provided via the selected monodirectional input data bus or a part of the output of that adder itself to the aforementioned input.

[0022]

The processing unit described in described in Claim 6 is based on Claim 2 and is characterized by the fact that the aforementioned input data bus selecting means includes a means which selects one of the aforementioned plural monodirectional input data buses or the

output of the aforementioned multiplier for at least one input of one of the aforementioned two adders in a controllable manner and provides the data via the selected monodirectional input data bus or a part of the output of the aforementioned multiplier to the aforementioned input.

[0023]

The processing unit described in described in Claim 7 is based on Claim 2 and is characterized by the fact that the aforementioned input data bus selecting means includes a means which, for at least one input of one of the aforementioned two adders, selects one of the aforementioned plural monodirectional input data buses or the output of that adder itself or a part of the output of the aforementioned multiplier in a controllable manner and provides the data via the selected monodirectional input data bus or the output of that adder itself or a part of the output of the aforementioned multiplier to the input.

[0024]

The processing unit described in described in Claim 8 is based on Claim 2 and is characterized by the following facts: the processing unit also includes a read only memory means used for storing prescribed information in advance; the aforementioned input data bus selecting means includes a means which selects one of the aforementioned plural monodirectional input data buses or the output of the aforementioned read only memory means for at least one input of the aforementioned multiplier and provides the data via the selected monodirectional input data bus or part of the output of the aforementioned memory means to the aforementioned input.

[0025]

The parallel processing processor described in Claim 9 is characterized by the following facts: the parallel processing processor has n processing units; first monodirectional data buses used to connect the adjacent processing units sequentially in a loop in a prescribed direction; second monodirectional data buses used to connect every other adjacent processing units sequentially bidirectionally in a loop; said n is a multiple of 4; each of the aforementioned processing units includes the following: plural operation means, which have plural inputs, respectively, that carry out a prescribed arithmetic operation between the provided data, and output the result; an input data bus selecting means, which is connected to the first monodirectional data bus that provides input from the adjacent processing unit and to the input data bus to the processing unit concerned out of the aforementioned second monodirectional data buses, that selects one of said first and second monodirectional data buses for each input of the aforementioned plural operation means, and provides part of the data provided via the selected monodirectional data bus to the aforementioned input; an output data bus selecting means, which

has an input connected to the outputs of the aforementioned plural operation means and an output connected to the aforementioned first monodirectional data bus that sends output to the adjacent processing unit and to the output data bus from the processing unit concerned out of the aforementioned second monodirectional data buses and outputs each output of the aforementioned operation means to one of the aforementioned first and second monodirectional data buses; and a control means, which is used to control the path of the data depending on the aforementioned input data bus selecting means and the output data bus selecting means in order to realize desired composite arithmetic operation by using the aforementioned plural operation means.

[0026]

The parallel processing processor described in Claim 10 is based on Claim 9 and is characterized by the following facts: the parallel processing processor also includes data storage means in the same number of the aforementioned processing units, each of which can output two data at a time as well as plural read data buses and write data buses used for connecting each of the aforementioned data storage means to each of the aforementioned processing units; in each processing unit, the aforementioned input data bus selecting means includes a means, which is connected to the first monodirectional data bus that provides input from the adjacent processing unit, to the input data bus to the processing unit concerned out of the aforementioned second monodirectional data buses, and to the aforementioned plural read data buses, that selects one data bus from said first and second monodirectional data buses and said read data buses for each input of the aforementioned plural operation means, and provides part of the data provided via the selected data bus to the aforementioned input; the aforementioned output data bus selecting means includes a means, which has an input connected to the outputs of the aforementioned plural operation means and an output connected to the aforementioned first monodirectional data bus that sends output to the adjacent processing unit, to the output data bus from the processing unit concerned out of the aforementioned second monodirectional data buses, and to the aforementioned write data buses and outputs each output of the aforementioned operation means to one of the aforementioned data buses.

[0027]

The parallel processing processor described in Claim 11 is based on Claim 9 and is characterized by the following facts: the parallel processing processor also includes data storage means in the same number of the aforementioned processing units, each of which can output two data at a time as well as plural read data buses and write data buses used for connecting each of the aforementioned data storage means to each of the aforementioned processing units; the

aforementioned plural processing units are divided into plural groups, each of which includes processing units in a number as a multiple of 4; one-to-one correspondence is established between the aforementioned plural processing units and the aforementioned plural data storage means; in each processing unit, the aforementioned input data bus selecting means includes a means, which is connected to the first monodirectional data bus that provides input from the adjacent processing unit, to the input data bus to the processing unit concerned out of the aforementioned second monodirectional data buses, and to read data bus from the data storage means corresponding to the processing unit concerned in the group including that processing unit among the aforementioned plural read data buses, that selects one data bus from said first and second monodirectional data buses and said read data buses for each input of the aforementioned plural operation means, and provides part of the data provided via the selected data bus to the aforementioned input;

the aforementioned output data bus selecting means includes a means, which has an input connected to the outputs of the aforementioned plural operation means and an output connected to the aforementioned first monodirectional data bus that sends output to the adjacent processing unit, to the output data bus from the processing unit concerned out of the aforementioned second monodirectional data buses, and to all of the aforementioned write data buses and outputs each output of the aforementioned operation means to one of the aforementioned data buses.

[0028]

The operation method described in Claim 12 is characterized by the following facts: the operation method is used to carry out desired arithmetic operation in a parallel processing processor including the following: four processing units; four first monodirectional data buses used to connect the adjacent processing units sequentially in a loop in a prescribed direction; and four second monodirectional data buses used to connect every other adjacent processing units bidirectionally; each processing unit includes the following: a multiplier, which has 2 n-bit inputs and multiplies the provided data and outputs a 2n-bit result; a first adder and a second adder, each of which has 2 n-bit inputs and adds the provided data and outputs an n-bit result; a carry switching means used to provide the carry output of the first adder to the carry input of the second adder in a controllable manner; an input data bus selecting means, which is connected to the aforementioned first monodirectional data bus that provides input from the adjacent processing unit and to the input data bus to the processing unit concerned from the aforementioned second monodirectional data buses, that selects the first or second monodirectional data bus for each input of the aforementioned plural operation means, and provides part of the data provided via the selected monodirectional data bus to the aforementioned inputs of the aforementioned multiplier and adders; an output data bus selecting

means, which has an input connected to the output of the aforementioned multiplier and the outputs of the aforementioned adders, to the aforementioned first monodirectional data bus that provides output to the adjacent processing unit, and to the output data bus from the processing unit concerned out of the aforementioned second monodirectional data buses and outputs each of the outputs of the aforementioned multiplier and adders to one of the first and second monodirectional data buses; and a control means, which controls the path of the data depending on the aforementioned input data bus selecting means and the output data bus selecting means in order to realize desired composite arithmetic operation by using the aforementioned multiplier and adders; the operation method has the following steps: the data needed for the operation are provided to the aforementioned input data bus selecting means; the aforementioned data are divided for every high-order n bits and low-order n bits by the aforementioned input data bus selecting means and are provided to any two inputs of the aforementioned multiplier and adders of the aforementioned four processing units; the aforementioned carry switching means of all of the processing units are set corresponding to the aforementioned desired arithmetic operation; the aforementioned output data bus selecting means and input data bus selecting means of each of the processing units are controlled to set the connection between the multiplier and the first and second adders of each processing unit in order to realize the aforementioned desired arithmetic operation; the aforementioned output data bus selecting means is controlled such that the prescribed part of the prescribed output among the outputs of the aforementioned multiplier and adders of the prescribed processing unit determined by the aforementioned desired arithmetic operation is output to the desired data bus so that the result of the arithmetic operation can be output to the desired data bus.

[0029]

The operation method described in Claim 13 is based on Claim 12 and is characterized by the fact that the step of setting the connection includes a step in which the data path depending on the output data bus selecting means and input data bus selecting means of the processing units is set such that the high-order n bits in the output of the multiplier or adders in a certain processing unit are input to the low-order n bits of the multiplier or adders in another processing unit.

[0030]

The operation method described in Claim 14 is based on Claim 12 and is characterized by the fact that the step of setting the connection includes a step in which the data path depending on the output data bus selecting means and input data bus selecting means of the processing units is set such that the high-order n bits in the output of the multiplier or adders in a certain

processing unit are output to the low-order n bits of the multiplier or adders in another processing unit.

[0031]

The parallel processing processor described in Claim 15 is based on Claim 9 and is characterized by also including plural instruction storage means, which are prepared for each processing unit and are used to store control instructions carried out by the aforementioned control means.

[0032]

The parallel processing processor described in Claim 16 is based on Claim 15 and is characterized by the fact that the aforementioned control instructions are classified into two systems, that is, a first type of control instructions used for controlling the corresponding data storage means and a second type of control instructions sued for controlling the aforementioned operation means by the aforementioned control means.

[0033]

The parallel processing processor described in Claim 17 is based on Claim 16 and is characterized by the fact that the first type of control instruction specifies the two reading addresses and one writing address of each data storage means.

[0034]

The parallel processing processor described in Claim 18 is based on Claim 16 and is characterized by the fact that the aforementioned control means, the aforementioned operation means, input data bus selecting means, and output data bus selecting means are used according to the previously provided second type of control instruction until the second type of instruction is changed.

[0035]

The parallel processing processor described in Claim 19 is based on Claim 16 and is characterized by the fact that the aforementioned instruction storage means includes an instruction memory used to store plural instructions, a program counter used to specify the reading address of the instruction memory, a means used to read two instructions at a time with the address specified by the aforementioned program counter as the header, a means used to determine whether the two read instructions belong to the same system, a means used to provide the control instructions to the aforementioned control means and/or data storage means based on

the judgment result, and a means used to increment the counted value of the aforementioned program counter by 1 or 2 based on the judgment result.

[0036]

Operation

For the processing unit used for parallel processing described in Claim 1, the number of monodirectional input data buses connected to the processing unit is the same as the number of monodirectional output data buses. Therefore, the data provided via any of these monodirectional input data buses can be processed by plural operation means, and the result can be output to any of the monodirectional output data buses. Since various operations can be carried out under the control of the control means, this processing unit is a general purpose processing unit, and the control of the control means is relatively simple. Also, since the numbers of input/output data buses are the same, it is easy to connect the processing units when manufacturing a parallel processing processor by combining plural said processing units. Since the processing units have the same configuration, interchangeability can be realized when controlling these processing units.

[0037]

The processing unit described in Claim 2 uses an nxn bit multiplier and 2 n+n-bit adders so that it is possible to carry out typical processing in an operation requiring a large amount data processing.

[0038]

For the processing unit described in Claim 3, one unit of high-order or low-order n bits of any monodirectional input data bus can be provided to each input of the operation means. Each output of these operation means can be output to the high-order n bits and/or low-order n bits of any monodirectional output data bus. Consequently, operation between 2n-bit data can be divided for the high-order n bits and the low-order n bits and can be carried out by using the nxn bit multiplier and n+n-bit adder.

[0039]

For the processing unit described in Claim 4, it is possible to select the case when the carry output of one of the two adders is used or not used as the carry input of the other adder. Consequently, it is possible to perform addition of 2n-bit data, addition of n bit data, and two different additions for n-bit data.

[0040]

For the processing unit described in Claim 5, for at least one input of one of the two adders, since one of the plural monodirectional input data buses or the output of that adder itself can be selected as the input data, it is possible to carry out an operation using the self-addition result, such as a sum-of-products operation, and an addition using data sent from a data bus.

[0041]

For the processing unit described in Claim 6, for at least one input of one of the two adders, since one of the plural monodirectional input data buses or the output of the multiplier can be selected as the input data, it is possible to carry out an operation using the multiplication result for the input of the addition, such as a sum-of-products operation, and an addition using data sent from a data bus.

[0042]

For the processing unit described in Claim 7, for at least one input of one of the two adders, since one of the plural monodirectional input data buses or the output of that adder itself, or part of the output of the multiplier can be selected as the input data, it is possible to carry out an operation using a multiplication result or an addition result for the input of addition, and an addition using data sent from a data bus.

[0043]

For the processing unit described in Claim 8, it is possible to select one of the monodirectional input data buses or the output of the read only memory means as the input data for at least one input of the multiplier. Consequently, it is possible to freely select the configuration used for efficiently performing high-speed operation by using the data stored in the read only memory means, such as division using the Newton-Raphson method and extraction-of-the-square-root operation, and the configuration used for performing normal operations.

[0044]

For the parallel processing processor described in Claim 9, n processing units are connected such that the adjacent processing units are connected monodirectionally by the first monodirectional data buses, while every other adjacent processing units are connected bidirectionally using the second monodirectional data buses. Each processing unit has the same number of inputs and outputs and has at least two inputs and two outputs. The degree of freedom of the operation increases as the number of inputs/outputs increases. Also, since the processing

units have the same configuration, the complexity in layout and control of the processor can be reduced.

[0045]

For the parallel processing processor described in Claim 10, a data storage means is prepared for each processing unit. Each processing unit can perform a prescribed processing using plural operation means to either the data provided from the data storage means or the data provided from another processor. The output of any processing unit can be output to another processing unit or to any data storage means.

[0046]

For the parallel processing processor described in Claim 11, the processing units are classified into groups, and a data storage means is prepared for each data storage means. Each processing unit can perform a prescribed processing using plural operation means to either the data provided from a data storage means corresponding to the processing unit belonging to the same group or to the data provided from another processor. The output of any processing unit can be output to another processing unit or to any data storage means. The read data bus from a data storage means to a processing unit belonging to the same group is short compared with the case when the processing units are connected without being grouped so that data can be provide at a high speed to the processing unit. Also, it is possible to carry out an operation using common data for a group of processing units. In addition, since data can be written from each processing unit into any data storage means, data exchange becomes possible between the groups.

[0047]

In the operation method for parallel processing processor described in Claim 12, data are divided into units of high-order and low-order n bits, which are provided to any two of the inputs of the multiplier and adders of the four processing units. The high-order n bits or low-order n bits of the computation result of another processing unit can be provided to the multiplier or adders as low-order n bits or high-order n bits of data used for new operation. Various operations can be performed. Also, since it is possible to select whether to input/output carry between two adders, it is possible to switch between two addition processing with n-bit precision and one addition processing with 2n-bit precision. Consequently, processing can be carried out with a very high degree of freedom.

[0048]

In the operation method for parallel processing processor described in Claim 13, the connection between the data buses and multiplier or adders can be set such that the high-order n bits or low-order n bits in the output of the multiplier in a certain processing unit can be input to either the low-order n bits or high-order n bits of the multiplier or adder in another processing unit. Various operations can be carried out without using a shift means.

[0049]

In the operation method for parallel processing processor described in Claim 14, the connection between the data buses and multiplier or adders can be set such that the high-order n bits in the output of the multiplier in a certain processing unit can be input to either the low-order n bits or high-order n bits of the multiplier or adder in another processing unit. Data can be well shifted without using a shift means, and there is no need to spend time for shifting.

[0050]

For the parallel processing processor described in Claim 15, control instructions are stored in an instruction storage means in each processing unit. Therefore, a complicated operation can be controlled for each processing unit, and the overall control of the parallel processing processor becomes easy.

[0051]

For the parallel processing processor described in Claim 16, control instructions are classified into two systems, that is, control instructions used for controlling the data storage means and control instructions used for controlling the operation means. When a certain instruction is repeated via the data output from the data storage means, if only the control instruction controlling the data storage means is issued repeatedly while changing its content, there is no need to issue several different control instructions used for the operation means.

[0052]

For the parallel processing processor described in Claim 17, a prescribed operation is carried out by reading two data from two read addresses of the data storage means. The result can be written at the position specified by one write address.

[0053]

For the parallel processing processor described in Claim 18, the operation means, the input data bus selecting means, and the output data bus selecting means are controlled based on

the previously received second type of control instruction until the second type of control instruction is changed. When a certain instruction is repeated via the data output from the data storage means, if only the control instruction controlling the data storage means is issued repeatedly while changing its content, there is no need to issue several different control instructions used for the operation means.

[0054]

For the parallel processing processor described in Claim 19, two instructions are read out at a time from the instruction memory with the address specified by the program counter used as the header, and it is determined whether the two instructions belong to the same system. If they belong to the same system, only the instruction read out first is executed, and the program counter is incremented by 1. If they do not belong to the same system, the control instructions are provided to both the control means and the data storage means, and the program counter is incremented by 2. Since the instructions of different systems can be executed at the same time, the instruction execution speed can be increased compared with the case when only one control instruction can be read out at a time.

[0055]

Application examples

First application example

The first application example of the present invention is a processor comprised of four processing units (PU). The processor shown in Figure 1 includes four processing units PU00, 01, 10, 11 (represented by 200, 202, 204, 206 in Figure 1) and monodirectional data buses 210, 212, 214, 216 used for connecting processing units 200, 202, 204, 206 sequentially in a loop. The numbers (00, 01, 10, 11) attached to processing unit PU represent the address of the processing unit in binary number.

[0056]

This processor also includes monodirectional input data buses 220, 222, 224, 226 used for connecting processing units, whose addresses differ from each other by 2, in a bidirectional manner. Said monodirectional data buses 210, 212, 214, 216 are used to connect the processing units, whose addresses different from each other by 1, in a sequential and monodirectional manner. Also, data bus 216 is the data bus from processing unit PU11 (206) to PU00 (200). In Figure 1, the arrow attached to each data bus indicates the direction of the data flow. In the processor shown in Figure 1, the number of inputs of the data buses going into a processing unit

is equal to the number of the outputs of the data buses coming out of a processing unit. This number is 2. This number is the same for all processing units.

[0057]

As shown in Figure 2, the processor disclosed in the first application example also includes control circuits (PUC) 250, 252, 254, 256, instruction memories (IM) 260, 262, 264, 266, and data memories 270, 272, 274, 276 corresponding to processing units 200, 202, 204, 206 besides said processing units PU00, 01, 10, 11 (200, 202, 204, 206).

[0058]

This processor also includes memory read bus group 280 used for the data read from data memories 270, 272, 274, 276 and memory write bus group 272 used for the data written from processing units 200, 202, 204, 206 into data memories 270, 272, 274, 276. Two data can be read out simultaneously from each of data memories 270, 272, 274, 276. Connection to different data buses in memory read bus group 280 can be realized by using data buses 300 and 201, 302 and 303, 304 and 305, 306 and 207. This connection will be described later. The data buses in memory read data bus group 280 are connected to processing units 200, 202, 204, 206 by data bus group 310, 312, 314, 316.

[0059]

Memory read data bus group 282 includes four n-bit data buses, which are connected to the outputs of processing units 200, 202, 204, 206 by data bus group 320, 322, 324, 326 comprised of 4 n-bit data buses. The details of this connect will be described later. One unit of data can be written to data memories 270, 272, 274, 276 simultaneously. The data written into the data memories is the computation result of each processing unit.

[0060]

Instructions provided from outside via an input/output port not shown in the figure are stored in instruction memories 260, 262, 264, 266. Each of control circuits 250, 252, 254, 256 has two outputs. One of the outputs is used to control data memories 270, 272, 274, 276, while the other output is used to control the computing elements in processing units 200, 202, 204, 206.

[0061]

Figure 3 shows the internal configuration of processing unit 200. The configuration of processing unit 200 shown in Figure 3 is an example. The other processing units 202, 204, 206

have exactly the same configuration as processing unit 200. Consequently, the explanation will not be repeated.

[0062]

As shown in Figure 3, memory write bus group 282 includes four memory write buses 400, 402, 404, 06. The data bus 210 to processing unit 202 shown in Figures 1 and 2 includes data bus 210M for high-order n bits and data bus 210L for low-order n bits. Similarly, the data bus 220 to processing unit 204 includes data bus 220M for high-order n bits and data bus 220L for low-order n bits.

[0063]

The memory read bus group 310 from memory written bus group 280 to processing unit 200 shown in Figure 2 includes data buses 330, 332, 334, 336, 340, 342, 344, 346 as shown in Figure 3. Memory read bus group 280 includes 8 data buses as described above. Said memory read buses 330, 332, 334, 336, 340, 342, 344, 346 are branched off from said 8 data buses, respectively. This connection will be described later based on Figure 4.

[0064]

Data bus 224 from processing unit 204 shown in Figures 1 and 2 includes data bus 224M for high-order n bits and data bus 224L for low-order n bits. Also, data bus 216 from processing unit 206 includes data bus 216M for high-order n bits and data bus 216L for low-order n bits.

[0065]

As shown in Figure 3, processing unit 200 includes selectors 350, 352, 354, 356 connected to read bus group 310, selectors 360, 362, 364, 366, 368, 370 with the outputs of selectors 350, 352, 354, 356 and data buses 216, 224, etc. connected to their inputs, multiplier (MPY) 380 with the outputs of selectors 360 and 363 connected to its inputs, a first adder (ADD0) 384 having two inputs connected to the outputs of selectors 368 and 370, a second adder (ADD1) 382 having two inputs connected to the outputs of selectors 364 and 366, crossbar switch (CBS) 392 that connects the n-bit outputs MPM and MPL of multiplier 380 to the outputs of adders 384, 382 and connects each output data to the data buses 210M and 220M for high-order n bits or the data buses 210L and 220L for low-order n bits of output data buses 210, 220, and selector (SELW) 390 that outputs the two outputs MPM, MPL of multiplier 380 and the outputs of two adders 382, 384 to one of the four memory write buses 400, 402, 404, 406 of memory write bus group 282. Selector 390 and memory write buses 400, 402, 404, 406 are connected by data buses 410, 412, 414, 416, respectively. Crossbar switch 392 is connected to

data buses 210, 220 by 2n-bit data buses 420, 422, respectively. The high-order n bits of data bus 420 are connected to data bus 210M, while the low-order n bits are connected to data bus 210L. The high-order n bits of data bus 422 are connected to data bus 220M, while the low-order n bits are connected to data bus 220L.

[0066]

Four groups of data buses 330, 332, 334, 336 are connected to the inputs of selectors 350 and 354. Four groups of data buses 340, 342, 344, 346 are connected to the inputs of selectors 352, 356.

[0067]

The output of selector 350 is connected to one of the inputs of selector 36. The data bus 224 for high-order n bits of data bus 224 is connected to the other input of selector 360. The output of selector 352 is connected to one of the inputs of selector 362. The data bus 216M for high-order n bits of data bus 216 is connected to the other input of selector 362.

[0068]

Each of selectors 364, 366 has three inputs. The output of selector 354 is connected to one of the inputs of selector 364. The data bus 224M for high-order n bits of data bus 224 is connected to another input of selector 364. The output AD1 of adder 382 is connected to the remaining input of selector 364. The output of selector 356 is connected to one of the inputs of selector 366. The high-order n bits MPM in the output of multiplier 380 are provided to another input of selector 366. The data bus 216M for high-order n bits of data bus 216 is connected to the remaining input of selector 366.

[0069]

Each of selectors 368, 370 has four inputs. The output AD0 of adder 384 is provided to the first input of selector 368. The output of selector 350 is provided to the second input. The data bus 224L for low-order n bits of data bus 224 is connected to the third input. The data bus 224M for high-order n bits of data bus 224 is connected to the fourth input. The first input of selector 370 is connected to the output of selector 352. The second input is connected to the data bus 216L for low-order n bits of data bus 216. The low-order n bits MPL of the output of multiplier MPY 380 are provided to the third input. The fourth input is connected to the output of selector 356.

[0070]

A carry output switch 386 is arranged between the carry output of adder 384 and the carry input of adder 382. Carry output switch 386 is opened/closed depending on control signal CC.

[0071]

As shown in Figure 4, the four data buses 400, 402, 404, 406 of memory write bus group 282 are connected to data buses 270, 272, 274, 276, respectively. On the other hand, memory read bus group 280 includes 8 memory read buses 290-297. Data memory 270 is connected to memory read buses 290, 291 by memory read buses 300, 301. Data memory 272 is connected to memory read buses 292, 293 by memory read buses 302, 303. Data memory 274 is connected to memory read buses 294, 295 by memory read buses 304, 305. Data memory 276 is connected to memory read buses 296, 297 by memory read buses 306, 307. Memory read buses 290-297 are branched and connected to processing units 200, 202, 204, 206 shown in Figure 2 as memory read bus group 310, 312, 314, 316.

[0072]

The following operations can be carried out by the processing unit having the configuration shown in Figure 3.

[0073]

(1) nxn bit multiplication, n+n-bit addition between the data read from the data memories.

[0074]

(2) nxn bit multiplication, n+n-bit addition between the data read from the data memories and the data provided from data bus 224M or 216M.

[0075]

(3) nxn bit multiplication, n+n-bit addition between the data on data buses 224M and 216M.

[0076]

(4) Addition between the high-order n bits (MPM) of the multiplication result output from multiplier 380 and the data on data bus 224M, addition between the low-order n bits (MPL) of the output of the multiplier MPY and the data on data bus 224.

[0077]

(5) Addition between the high-order n bits (MPM) of the multiplication result and the data read from the data memories, addition between the low-order n bits (MPL) of the multiplication result and the data read from the data memories.

[0078]

(6) Addition between the high-order n bits (MPM) of the multiplication result and the addition result of adder 382, addition between the low-order n bits (MPL) of the multiplication result and the addition result of adder (384) (sum-of-products operation).

[0079]

(7) Addition between the 2n-bit number expressed by data buses 224M and 224L and the 2n-bit number expressed by data buses 216M and 216L.

[0080]

The control system for the parallel processing processor of the first application example shown in Figures 1-4 is explained below. In the parallel processing processor of the first application example, each processing unit is controlled independently. There are instruction memories 260, 262, 264, 266 corresponding to the processing units (see Figure 2).

[0081]

The instructions of each processing unit are classified into at least two systems, that is, data memory control instruction used for controlling the data memories and the computing element control instructions used for controlling the computing elements in the processing unit. The data memory control instructions are output from control circuits 250, 252, 254, 256 shown in Figure 2 to the buses connected to the corresponding data memories 290, 292, 294, 296. The computing element control instructions are output from controls circuits 250, 252, 254, 256 to the buses going to corresponding processing units 200, 202, 204, 206.

[0082]

Figure 5(a) shows the form of data memory control instruction 430. Data memory control instruction 430 includes OP field 432, src0, scr1 fields 434, 436, and dst field 438.

[0083]

OP field 432 is used to designate the address mode. Scr0, scr1 fields are used to designated the two addresses of the data read from the corresponding data memory. The dst field 438 is used to designate the address of the data written into the data memory and the connection between the outputs of the computing elements and memory write buses 400, 402, 404, 406.

[0084]

Figure 5(b) shows the form of computing element control instruction 450. Computing element control instruction 450 includes MPY field used for controlling the multiplier, ADD1 field, ADD0 field used for controlling adders ADD1, ADD0, and SEL field storing the data used for selecting four data from 8 memory read buses 290-297.

[0085]

MPY field includes OP0 field 452, scr00 field 454, scr01 field 456, and dst0 field 458. ADD1 field includes OP1 field 460, src10 field 462, src11 field 464, dst1 field 466. ADD0 field include OP2 field 468, src20 field 470, src21 field 472, dst2 field 474.

[0086]

OP0 field 452, OP1 field 460, and OP2 field 468 are used to designate the operation contents of the corresponding computing elements. Scrc00 field 454, src01 field 456, src10 field 462, src11 field 464, src20 field 470, src21 field 472 store the data used for controlling the selectors set at the input of each computing element. dst0 field 458, dst1 field 466, dst2 field 474 store data used for controlling crossbar switch CBS 392 (Figure 3) used for connecting the computing elements and the data buses.

[0087]

SEL field includes SEL0 field 476, SEL1 field 478, SEL2 field 480, and SEL3 field 482. Each field stores data used for selecting one from 8 memory read buses. Consequently, four data are selected by the SEL field.

[0088]

Among the fields shown in Figure 5(b), dst0 field 458 is further divided into two fields dst00 field and dst01 field not shown in the figure. They are used to store data for designating the output destinations of the two outputs MPM and MPL of the multiplier MPY.

[0089]

The values of the data stored in fields 454, 456, 458 (the aforementioned two fields dst00, dst01) and fields 462, 464, 466, 470, 472, 474 shown in Figure 5(b) and the selecting operations of each selector corresponding to each value are shown in the following Tables 1-10.

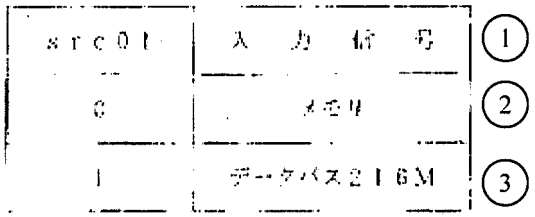
[0090]

Table 1

[First table]

Key: 1 Input signal
2 Memory
3 Data bus 224M

[Second table]



Key: 1 Input signal
2 Memory
3 Data bus 216M

[0091]

Table 2

[Third table]

dst00	出力先	①
00	データバス210M	②
01	データバス210L	
10	データバス220M	
11	データバス220L	

Key: 1 Output destination
2 Data bus

[Fourth table]

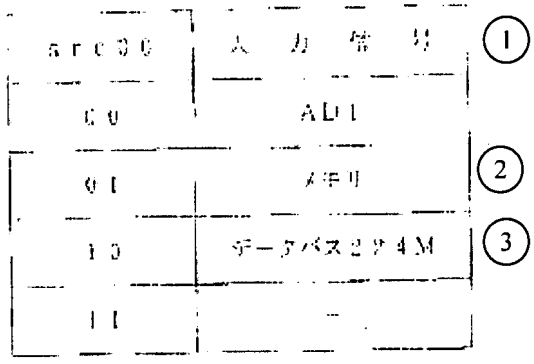
dst01	出力先	①
00	データバス210M	②
01	データバス210L	
10	データバス220M	
11	データバス220L	

Key: 1 Output destination
2 Data bus

[0092]

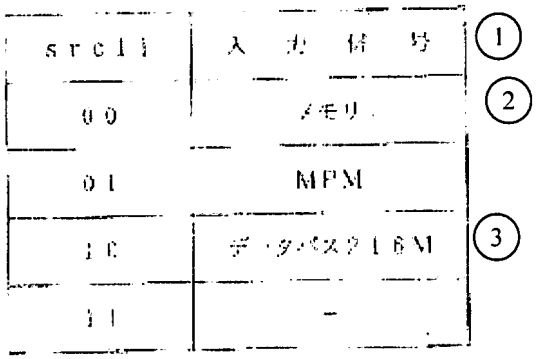
Table 3

[Fifth table]



Key: 1 Input signal
2 Memory
3 data bus 224M

[Sixth table]



Key: 1 Input signal
2 Memory
3 Data bus 216M

[Seventh table]

dst1	出力先 ①
00	データバス210M
01	データバス210L
10	データバス220M
11	データバス220L

② (01, 10, 11)

Key: 1 Output destination
2 Data bus

[0093]

Table 4
[Eighth table]

src20	入力信号 ①
00	AD0
01	メモリ ②
10	データバス224L
11	データバス224M

③ (10, 11)

Key: 1 Input signal
2 Memory
3 Data bus

[Ninth table]

src2	入力信号	①
00	メモリ	②
01	MPM	
10	データバス216M	③
11	—	

Key: 1 Input signal
2 Memory
3 Data bus 216M

[Tenth table]

dst2	出力先	①
00	データバス210M	②
01	データバス210L	
10	データバス220M	
11	データバス220L	

Key: 1 Output destination
2 Data bus

Dst field 438 shown in Figure 5(a) is further divided into three fields, that is, field PU selecting field 440, SELW control field 442, and write address designation field 444. Among these fields, fields 440, 442 have a 2-bit length. The values of the data stored in said fields 440, 442, the processing unit selected by each value, and the list of the output of each computing element selected by SELW are shown in the following Tables 11 and 12.

[0094]

Table 5

[Eleventh table]

① PU選択フィールド	選択PU ②
0 0	PU 0 0
0 1	PU 0 1
1 0	PU 1 0
1 1	PU 1 1

Key: 1 PU selecting field
2 Selected PU

[Twelfth table]

① SELW制御フィールド	選択出力 ②
0 0	MPM
0 1	MPL
1 0	AD1
1 1	AD0

Key: 1 SELW control field
2 Selected output

Control circuits 250, 252, 254, 256 shown in Figure 2 control each selector as shown in the first-twelfth tables according to the instructions shown in Figures 5, 6.

[0095]

Figure 6 shows the form of storing instructions in instruction memories 260, 262, 264, 266 shown in Figure 2. In Figure 6, "MCNT" means data memory control instruction, while "PCNT" means computing element control instruction. Instruction memory 260 basically stores the aforementioned data memory control instruction and the computing element control

instruction as a group as shown by instructions 490 and 492. The numbers (100, 101, 102, 103) shown on the left side of instruction memory 216 in Figure 6 indicate the addresses of instruction memory 260. In the example shown in Figure 6, MCNT instruction 490 is stored at address 100 and PCNT instruction 492 is stored at address 101. When the program counter address points to "100," CNT instruction 490 at address 100 will be read out as the next instruction.

[0096]

The instructions are processed as described above with data memory control instruction MCNT and computing element control instruction PCNT as a group. However, when the same operation is repeated for a large amount of data, it is only necessary to set the input conditions and the operation content once at the beginning for each computing element. After that, the processing can be carried out by sequentially changing the address of the data memory indicating the position of the data. In this case, as indicated by addresses 102, 103 in Figure 6, MCNT instructions 494, 496 are stored consecutively in instruction memory 260. Each computing element carries out the same operation content repeatedly based on the computing element control instruction set in the previous round until receiving the next computing element control instruction PCNT.

[0097]

In the following, the operation of the parallel processing processor disclosed in the first application example will be explained based on specific examples. In the following example, the selectors and crossbar switch in each processing unit are switched appropriately to provide desired connections under the instructions set appropriately to provide the connection shown in each diagram according to the first-twelfth tables.

[0098]

Figure 7 shows an example, in which each of processing units 200, 202, 204, 206 carries out an n-bit precision operation independently. In this case, the data buses for connecting the processing units are not used. In each of processing units PU00-PU10 (240, 242, 244, 246), "X" means a multiplier, while "+" means an adder.

[0099]

In processing unit 200, both of the inputs of the multiplier are the data read from the data memory. The multiplier has an nxn-bit configuration. Its output has 2n bits. Selector SELW 390 shown in Figure 3 is set such that after the high-order n bits or 2n bits in the output of the multiplier are rounded off, the high-order n bits are output to the data memory. Although a

special hardware for rounding off is needed, since it has no direct relationship to this patent application, it will not be explained.

[0100]

In processing units 202, 204, the data read from the data memory are provided to the two inputs of one of the adders. In other words, the selection in the input part of each adder is set appropriately to select the data read from the data memory. In processing unit 206, a sum-of-products operation is carried out. In other words, the two data read from the data memory are provided to the two inputs of the multiplier. The high-order n bits in the output of the multiplier are provided to one of the inputs of the adder, and the output of the adder is provided to the other input of that adder itself.

[0101]

In the connection example shown in Figure 7, the output from each processing unit is written into the data memory via memory write buses 400, 402, 404, 406 (see Figure 3). This is the same in other connection examples to be described later. Selector SELW 390 of each processing unit is controlled appropriately so that the output of the multiplier or adder, from which the desired computation result is obtained, is written to the desired memory write bus among memory write buses 400, 402, 404, 406.

[0102]

Figure 8 shows the connection relationship of the data buses of the parallel processing processor in this application example when carrying out multiplication with a double precision of 2n bits. The data used as the multiplication objects are taken as a, b. The high-order n bits and low-order n bits of data a are represented by a1, a0, respectively. The high-order n bits and low-order n bits of data b are represented by b1, b0, respectively. Multiplication "axb" is expressed as follows.

[0103]

(a0+a1)x(b0+b1)

=a0xb0 + a0xb1 + a1xb0 + a1xb1

In other words, multiplication axb between 2n-bit numbers can be divided into the sum of four multiplications between n-bit numbers, that is, a0xb0, a0xb1, a1xb0, a1xb1. The connection example shown in Figure 8 is used to carry out the aforementioned calculation after dividing the two 2n-bit numbers a, b into high-order n bits and low-order n bits.

[0104]

In the following, the connection relationship in each processing unit will be explained. In each processing unit, the adder on the left side is ADD1, while the adder on the right side is ADD0.

[0105]

In processing unit 200, each selector is set appropriately so that the two data read from the data memory are provided to the two inputs of the multiplier. The high-order n bits out of the 2n bits in the output of the multiplier are connected to one of the inputs of adder ADD0 in processing unit 202 via data bus 201L. The output of adder ADD1 of processing unit 206 is connected to the other input via data bus 216M. The output of adder ADD0 in processing unit 200 is connected to one of its own inputs. The output of adder ADD0 in processing unit 206 is connected to the other input via data bus 216L. The carry C from adder ADD0 is provided to the carry input of adder ADD1.

[0106]

In processing unit 202, the selectors are set appropriately so that the two data read from the data memory are provided to the two inputs of the multiplier. The high-order n bits out of the 2n bits output from the multiplier are provided to one of the inputs of adder ADD1 in processing unit 202. Constant "0" is provided to the other input of adder ADD1. The output of multiplier [sic, adder] ADD1 in processing unit 202 is connected to one of the inputs of adder ADD1 in processing unit 204 via data bus 212M. The low-order n bits of the multiplier in processing unit 202 are provided to one of the inputs of adder ADD0 in processing unit 202. The high-order n bits of the output of the multiplier in processing unit 200 are provided to the other input as described above. In processing unit 202, carry output C of adder ADD0 is also provided to adder ADD1.

[0107]

In processing unit 204, the two data read from the data memory are provided to the two inputs of the multiplier. The high-order n bits among the 2n-bit output of the multiplier are connected to one of the inputs of adder ADD1, and the low-order n bits are connected to one of the inputs of adder ADD0. The other input of adder ADD0 is connected to the output of adder ADD0 in processing unit 202 via data bus 212L. The carry output C of adder ADD0 is provided to adder ADD1. The output of adder ADD1 is connected to one of the inputs of adder ADD0 of processing unit 206 via data bus 214L.

[0108]

In processing unit 206, the selectors are set appropriately so that the two data read from the data memory are provided to the two inputs of the multiplier. The high-order n bits in the output of the multiplier are provided to one of the inputs of adder ADD1. The low-order n bits are connected to one of the inputs of adder ADD0. Constant 0 is provided to the other input of adder ADD1. The output of adder ADD1 of processing unit 204 is connected to the other input of adder ADD0 via data bus 214L. The output of adder ADD0 is connected to one of the inputs of adder ADD0 of processing unit 200 via data bus 216L. The output of adder ADD1 is connected to one of the inputs of adder ADD1 of processing unit 200 via data bus 216M. The carry output of adder ADD0 is provided to the carry input of adder ADD1.

[0109]

For the connection shown in Figure 8, the operation is started from multiplication of the least significant bit array. Said data a0 and b0 are provided from data memory to the multiplier in processing unit 200. Said data a1 and b0 are provided to the multiplier in processing unit 202 via the data memory. Said data a0 and b1 are provided from the data memory to the multiplier in processing unit 204. Data a1 and b1 are provided from the data memory to the multiplier in processing unit 206.

[0110]

The high-order n bits of the multiplication result in processing unit 200 are provided to adder ADD0 of processing unit 202 via data bus 210L. Data bus 210L is used for the low-order n bits of data bus 210. Outputting the high-order n bits of the multiplication result to the low-order n bits of the data bus is virtually equivalent to shifting the data to lower order by n bits.

[0111]

The two adders ADD0, ADD1 Of processing unit 202 performs 2n-bit addition between the multiplication result a1xb0 of the multiplier and a0xb0 obtained by shifting the data to lower order by n bits. The high-order n bits of the addition result are provided to processing unit 204 via data bus 212M, and the low-order n bits are provided to processing unit 204 via data bus 212L. In other words, the data are not shifted in this case.

[0112]

For the selectors in the input part of the multiplier of processing unit 204, the connection is set such that data a0 and b1 read from the data memory are provided to the input of the

multiplier. The high-order n bits of the output of the multiplier are provided to adder ADD1, while the low-order n bits are provided to adder ADD0. Since carry input/output is performed between the two adders of processing unit 204, adders ADD0 and ADD1 performs 2n-bit addition by adding a0xb1 to the 2n-bit data output from processing unit 202.

[0113]

For the selectors set in the input part of the multiplier of processing unit 206, the connection is set such that data a1 and ab1 from the data memory are provided to the two inputs of the multiplier. The high-order n bits of the output of the multiplier are provided to adder ADD1. The low-order n bits are provided to adder ADD0. The high-order n bits of the result of the addition performed in processing unit 204 are shifted to lower order by n bits and are provided to adder ADD0 in processing unit 206 via data bus 214L for the low-order n bits of data bus 214. Consequently, processing unit 206 performs 2n-bit addition by adding a1xb1 to the output of processing unit 206 shifted to the lower order by n bits.

[0114]

As described above, the multiplication result of axb between 2n bits is obtained at the output of processing unit 206.

[0115]

In the case of performing a sum-of-products operation, the outputs of adders ADD0 and ADD1 in processing unit 206 shown in Figure 8 are provided to adders ADD0 and ADD1 of processing unit 200 via data buss 216L and 216M, respectively.

[0116]

In the connection example shown in Figure 8, all of the multipliers and the adders in processing units 202, 204, 206 are needed to perform a multiplication with a 2n-bit precision. At the same time, 2n-bit precision operation can also be performed using the adders in processing unit 200. Consequently, in this connection example, the processor can carry out one round of multiplication with a 2n-bit precision and one round of addition with a 2n-bit precision at the same time. Figure 9 shows a connection example in the case of performing a butterfly operation used for FTT (high-speed Fourier transformation) in the processor of this application example. The operation has an n-bit precision. In the butterfly operation, operations expressed as c+axb and c-axb are carried out between three complex numbers a, b, c. If ar, br, cr represent the real numbers of a, b, c, ai, bi, ci represent the imaginary numbers, and j represents the imaginary number unit, a, b, c are expressed as follows.

[0017]

a=ar+j.ai

b=br+j.bi

c=cr+j.ci

c+axb and c-axb can be calculated in the following four formulas by combining the real numbers and the imaginary numbers.

[0118]

$$cr + (ar \times br - ai \times bi) \quad \cdots (1)$$

$$ci + (ar \times bi + ai \times br) \quad \cdots (2)$$

$$cr - (ar \times br - ai \times bi) \quad \cdots (3)$$

$$ci - (ar \times bi + ai \times br) \quad \cdots (4)$$

In appearance, it is necessary to carry out four operations in order to derive each of formulas (1)-(4). However, since these formulas have common items, only 4 multiplications and 6 additions (or subtractions) are actually needed. The connection for carrying out this butterfly operation is shown in Figure 9.

[0119]

As shown in Figure 9, in processing unit 200, two data read from the data memory are provided to the two inputs of the multiplier. The high-order n bits in the input of the multiplier are provided to one of the inputs of the adder ADD1 in processing unit 200, while data bus 224M is connected to the other input. The output of adder ADD1 is connected to data bus 220M.

[0120]

Two data read from the data memory are provided to the two inputs of the multiplier in processing unit 202. The high-order n bits in the output of the multiplier are connected to one of the inputs of adder ADD1 in processing unit 202, while data bus 226M is connected to the other input. The output of adder ADD1is connected to data bus 222M.

[0121]

Two data read from the data memory are provided to the two inputs of the multiplier in processing unit 204. The high-order n bits of the output of the multiplier are connected to processing unit 200 via data bus 224M. The data from the data memory are provided to the other

input of adder ADD1. The inputs of adder ADD0 are also common connected to the inputs of adder ADD1.

[0122]

Two data read from the data memory are provided to the two inputs of the multiplier in processing unit 206. The high-order n bits of the output of the multiplier are connected to processing unit 202 via data bus 226M. One of the inputs of two adders ADD0, ADD1 in processing unit 206 are commonly connected to processing unit 202 via data bus 222M. The data from the data memory are commonly provided to the other inputs.

[0123]

For both processing units 204 and 206, addition is performed in one of adders ADD0 or ADD1, while subtraction is performed in the other adder.

[0124]

In the processor connected as shown in Figure 9, when data are provided as follows, cr+(ar.br-ai.bi) and cr-(ar.br-aibi), ci+(ar.bi+ai.br) and ci-(ar.bi+ai.br) are obtained as the outputs of the adders in processing units 204 and 206, respectively.

[0125]

ar, br read from the data memory are provided to the two inputs of the multiplier in processing unit 200. ar, bi read from the data memory are provided to the two inputs of the multiplier in processing unit 202. ai, bi read from the data memory are provided to the two inputs of the multiplier in processing unit 204. cr read from the data memory is provided to one of the inputs of two adders ADD0, ADD1 in processing unit 204. ai, br erad from the data memory are provided to the two inputs of the multiplier in processing unit 206. ci read from the data memory is provided to one of the inputs of two adders ADD0 and ADD1 in processing unit 206.

[0126]

ai.bi is provided via data bus 224M to processing unit 200 from the multiplier o fprocessing unit 204. ar.br-aibi is provided via data bus 220M to processing unit 204 from the adders in processing unit 200. The addition and subtraction between ci and ar.br-ai.bi are performed by the two adders in processing unit 204, respectively. Consequently, said formulas (1) and (3) are obtained at the two outputs of the adders as described above.

[0127]

ai.br is provided from the multiplier in processing unit 206 to processing unit 202 via data bus 226M. The adder ADD1 in processing unit 202 outputs ar.bi+ai.br. The output is provided to the two adders in processing unit 206 via data bus 222M. Addition is performed between ci and ar.bi+ai.br by one of the two adders in processing unit 206, while subtraction is performed by the other adder. Consequently, said formulas (2) and (4) are obtained as the outputs of the two adders in processing unit 206.

[0128]

In this example, the 2n-bit data output from the multipliers of processing unit s206, 204 are rounded off to n bits by an appropriate rounding-off operation before they are input into the adders.

[0129]

Figure 10 shows the data bus connection between the processing units in the processor when performing a sum-of-products operation with an n-bit precision. The operation of adding ai.bi while changing i can be performed in a unit of four items since the processor includes four processing units. First, the connection example of the processor shown in Figure 10 will be explained.

[0130]

Two data read from the data memory are provided to the two inputs of the multiplier in processing unit 200. The high-order n bits of the output of the multiplier in invention unit 200 are provided to one of the inputs of adder ADD1. The other input of adder ADD1 is connected to data bus 224M. The output of adder ADD1 is connected to processing unit 204 via data bus 220M.

[0131]

Two data read from the data memory are provided to the two inputs of the multiplier in processing unit 202. The output of the multiplier is connected to one of the inputs of adder ADD1. Data bus 226M is connected to the other input of adder ADD1. The output of adder ADD1 is connected to processing unit 204 via data bus 212M.

[0132]

Two data read from the data memory are provided to the two inputs of the multiplier in processing unit 204. The high-order n bits in the output of the multiplier are connected to one of

the inputs of adder ADD1 of processing unit 200 via data bus 224M. One of the inputs of adder ADD1 Of processing unit 204 is connected to data bus 12M, and the other input is connected to data bus 220M. The output of adder ADD1 is connected to processing unit 206 via data bus 214M.

[0133]

Two data read from the data memory are input to the two inputs of the multiplier of processing unit 206. The high-order n bits in the output of the multiplier are connected to one of the inputs of adder ADD1 of processing unit 202 via data bus 226M. One of the inputs of adder ADD1 of processing unit 206 is connected to processing unit 204 via data bus 214M. The other input is connected to the output of adder ADD1 itself.

[0134]

In the processor connected as shown in Figure 10, a sum-of-products operation with n-bit precision is performed as follows.

[0135]

(a0, b0), (a1,b1), (a2, b2), (a3, b3) are provided from the respective data memories to processing units 200, 202, 204, 206. a0b0, a1b1, a2b2, a3b3 are obtained as the outputs of the multipliers in processing units 200, 202, 204, 206.

[0136]

a0b0+a2b2 is obtained as the output of processing unit 200, and a1b1+a3b3 is obtained as the output of processing unit 202. They are added by adder ADD1 in processing unit 204 to obtain a0b0+a1b1+a2b2+a3b3. In this example, the 2n bits of the output of each multiplier are rounded off to n bits by an appropriate rounding-off operation before they are input into the adders.

[0137]

Figure 11 shows a connection example in the case of performing a sum-of-products operation with n-bit precision using another method in the processor of this application example. In the data bus connection shown in Figure 10, a sum of products is obtained every four items. In the connection shown in Figure 11, ai+1, bi+1 are input after delayed by 1 clock with respect to ai, bi. As a result, the result of adding one item a time is obtained as the output of each processing unit. The final result is obtained at the output of the adder in processing unit 200 for the sum of products of every 4 items.

[0138]

The connection shown in Figure 11 is as follows. In processing unit 200, two data (a0, b0) read from the data memory are provided to the two inputs of the multiplier. The high-order n bits of the output of the multiplier are connected to one of the inputs of adder ADD1 in processing unit 202. One of the inputs of adder ADD1 in processing unit 200 is connected to data bus 216M. The other input is connected to the output of adder ADD1 itself.

[0139]

In processing unit 202, two data (a1, b1) read from the data memory are provided to the two inputs of the multiplier. The high-order n bits in the output of the multiplier are connected to one of the inputs of adder ADD1 in processing unit 202. The other input of adder ADD1 is connected to the high-order n bits of the output of the multiplier in processing unit 200. The output of adder ADD1 is connected to data bus 212M.

[0140]

In processing unit 204, two data (a2, b2) read from the data memory are provided to the two inputs of the multiplier. The high-order n bits of the output of the multiplier are connected to one of the inputs of adder ADD1 of processing unit 204. The other input of adder ADD1 is connected to the output of adder ADD1 of processing unit 202 via data bus 212M. The output of adder ADD1 in processing unit 204 is connected to data bus 214M.

[0141]

Two data (a3, b3) read from the data memory are provided to the two inputs of the multiplier in processing unit 206. The high-order n bits of the output of the multiplier are connected to one of the inputs of adder ADD1 of processing unit 206. The other input of adder ADD1 is connected to processing unit 204 via data bus 214M. The output of the adder ADD1 in processing unit 206 is connected to one of the inputs of adder ADD1 in processing unit 200 via data bus 216M.

[0142]

In the processor connected as shown in Figure 1, the following sum-of-products operation is carried out. First, a0b0 is obtained as the output of the multiplier in processing unit 200. Then, a0b0+a1b1 is obtained as the output of the adder in processing unit 202. Then, a0b0+a1b1+a2b2 is obtained as the output of the adder in processing unit 204. Also, a0b0+a1b1+a2b2+a3b3 is obtained as the output of processing unit 206.

[0143]

In this example, the 2n bits of the output of the multiplier in each processing unit are rounded off to n bits by an appropriate rounding-off operation before they are input into the adders.

[0144]

In the processor disclosed in the aforementioned first application example, a wide range of operations can be carried out by appropriately switching the selectors in each processing unit. Since the processing units have exactly the same configuration, the layout and connection relationship in the processor are simple. Also, since the processing units have the same configuration, the control instructions for controlling these processing units are interchangeable, and the processor can be controlled easily. Also, 2n-bit width data buses can be used between the processing units to provide the high-order n bits in the multiplication result as the low-order n bits of the next operation to another processing unit. Consequently, data can be shifted by n bits without using a data shifting means so that a variety of operations can be carried out using a simple circuit. Since the shift processing is unnecessary, the processing speed can be increased.

[0145]

Second application example

Figure 12 shows one (PU00) of the processing units used for the processor disclosed in the second application example. This processing unit 520 is different from processing unit 200 disclosed in the first application example shown in Figure 3 in the fact that it also includes a ROM (read only memory) 530 that receives the address from selector 350 and outputs the data at that address to the input of selector 360. The rest of the configuration of processing unit 520 is exactly the same as processing unit shown in Figure 30. Consequently, the rest of the configuration will not be explained in detail.

[0146]

If the processor includes four of said processing units, the other three processing units (PU01, PU10, PU11) have exactly the same configuration as said processing unit 520.

[0147]

The processing unit 520 in the processor disclosed in the second application example shown in Figure 12 can also perform the following operation besides the operation of processing unit 200 of the first application example shown in Figure 3. The address of ROM 530 is input

from the data memory into processing unit 520. The address signal is selected by selector 350 and is applied to ROM 530. ROM 530 provides the data stored at the designated address to selector 360. Selector 360 selects that data and provides it to one of the inputs of multiplier 380.

[0148]

Data used for division performed using the Newton-Raphson method or for extraction-of-the-square-root operation are stored in ROM 530. For the division performed using the Newton-Raphson method, first, the reciprocal of the divisor is derived by a recurrence formula depending on multiplication and addition. Finally, the dividend is divided by that reciprocal to derive the solution. In this case, it is a well-known fact that if the initial approximate value is not a sufficiently close value when deriving the reciprocal using the recurrence formula, the convergence of the recurrence formula becomes poor. This approximate value is stored in the ROM in advance. When it is used as the initial approximate value for the operation, the convergence of the recurrence formula can be improved significantly. The aforementioned operation can be carried out efficiently.

[0149]

Third application example

Figure 13 shows the main parts of the third application example of the processor disclosed in the present invention. The processor of the third application example executes instructions in a different way from the first application example. In the third application example, instruction memory 542 shown in Figures 13 and 14 is used instead of instruction memory 260 (see Figure 2) used in the first application example. Figures 13 and 14 only shows instruction memory 542 used for controlling processing unit 200. Other instruction memories used for controlling other processing units have exactly the same configuration as instruction memory 542.

[0150]

As shown in Figure 13, processing unit of this processor is directly controlled by control circuit 540. Control circuit 540 controls processing unit 200 according to the computing element control instruction provided from instruction memory 542. Instruction memory 542 is also connected to data memory 270. It is used to provide the data memory control instruction directly to data memory 270 without going through control circuit 540.

[0151]

As shown in Figure 14, instruction memory 542 includes memory 550 used for storing plural instructions, instruction register 552 used for storing two instructions read from memory 550, exclusive OR (EXOR) circuit 556 having two inputs connected to instruction register 552, switch 554 that is controlled by the output of EXOR circuit 556 and is used to select whether to output the second instruction out of the two instructions stored in instruction register 552 to control circuit 540 (Figure 13), address computing logic element 558 used for calculating the address of the instruction to be read next from memory 550 according to prescribed logic, and program counter (PC) 56) used for designating the two consecutive read addresses in memory 550 according to the computation result of address computing logic element 558.

[0152]

The numbers (100, 101, 102, 103) shown on the left side of memory 550 in Figure 14 indicate the addresses where the instructions are stored. The header (first bit) 570 of each instruction 572 stored in memory 550 is a flag indicating whether the instruction is a computing element control instruction or a data memory control instruction. In the example shown in Figure 14, if the first bit 570 is "0," the instruction is a memory control instruction. If it is "1," the instruction is a computing element control instruction.

[0153]

Similarly, instruction register 552 also has a region for storing instruction 582 and a region for storing the first bit 580. Instruction register 552 has two regions for storing these instructions, and the first bits are connected to the two inputs of EXOR circuit 556, respectively.

[0154]

Instruction memory 542 in the third application example shown in Figure 14 is used to further improve the control on the processor described in the first application example. In the processor of the first application example, the instructions are output one at a time from the instruction memory. After it is determined whether the instruction is a data memory control instruction or a computing element control instruction, the data memory or the computing element of each processing unit is controlled based on the identification result. However, the data memory and the computing element can be operated (controlled) independently from each other. Consequently, when a data memory control instruction and a computing element control instruction are stored consecutively in the instruction memory, these two instructions are executed at the same time. However, the efficiency is better if they are executed sequentially.

The third application example is used to improve the control method of the processor with respect to this fact.

[0155]

As shown in Figure 14, pointer 1 (PC+A) of program counter 560 points to address 100, while pointer 2 (PC+A+1) points to address 101. The instructions at addresses 100 and 101 are read into instruction register (IReg) at the same time. The first bits 580 of the two instruction stored in instruction register 552 are provided to the two inputs of EXOR circuit 556, respectively. The output of EXOR circuit 556 is "0" if the two first bits 580 stored in instruction register 552 are both "1" or "0." If the header bit of the instruction is set to "0" for data memory control instruction and to "1" for computing element control instruction, it is able to determine whether the two instructions are of the same system depending on the output of the EXOR circuit.

[0156]

If the output of the EXOR circuit is "1," switch 554 is closed. In this case, the second instruction (PCNT instruction in the example shown in Figure 14) is sent to control circuit 540 (Figure 13). If the output of EXOR circuit 556 is 0, switch 554 is open, and the second instruction is not sent to control circuit 540. On the other hand, the first instruction is constantly sent to the data memory.

[0157]

If the output of EXOR circuit 556 is "1," 2 is substituted into the A of program counter 560 by address computing logic element 558. If the input is "0," 1 is substituted to A and is provided to program counter 560. In other words, if instruction of different systems are read out, the pointer 1 (PC+A) of program counter 560 points to the next address 102 in memory 550. Pointer 2 (PC+A+1) points to address 103. Consequently, the instruction at address 101 will not be changed or read.

[0158]

When instructions of the same system are read out, pointer 1 (PC+A) points to address 101, and pointer 2 (PC+A+1) points to address 102. In this case, the instructions at addresses 101 and 102 are read out at the same time and are stored in instruction register 552. Then, the instructions are identified as described above, and transfer of the instructions and address computation are carried out based on the identification result.

[0159]

In the third application example, instructions are always read out two at a time. If they are of the same system, the second instruction is not executed, the program counter is incremented by 1, and two instructions including the second instruction are read out next. If the instructions are of different systems, the two instructions are executed at the same time, the program counter is incremented by 2, and the next two instructions after the aforementioned second instruction are read out next. When a data memory control instruction and a computing element control instruction are stored consecutively, they can be executed at the same time to improve the operation efficiency of the processor. Also, since each computing element operates according to the computing element control instruction input previously, the same operation can be carried out efficiently with respect to different data by providing the control instruction in the data memory repeatedly.

[0160]

Fourth application example

Figure 15 shows the processor disclosed in the fourth application example of the present invention. The processor shown in Figure 15 includes 8 processing units 600, 602, 604, 606, 608, 610, 612, 614 and 8 data memories 620, 622, 624, 626, 628, 630, 632, 634. The number attached to each processing unit indicates the address of that processing unit expressed in binary number. The number attached to data memory (DM) also indicates the address.

[0161]

This processor also include memory write bus group 680 connected to data memories 620, 622, 624, 626, 628, 630, 632, 634 by data bus group 690, 692, 694, 696, 698, 700, 702, 704, 706, memory read bus group 682 that is connected to data memories 620, 622, 624, 626 and is connected to processing units 600, 602, 604, 606 by data bus group 710, 712, 714, 716, and memory read bus group 684 that is connected to data memories 628, 630, 632, 634 and is connected to processing units 608, 610, 612, 614 by data bus group 718, 720, 722, 724.

[0162]

Also, as described in the first application example, this processor includes data buses 640, 642, 644, 646, 648, 650, 652, 654 that connect the processing units whose addresses differ by 1 monodirectionally and sequentially in a loop as well as data buses 660, 662, 664, 666, 668, 670, 672, 674 that connect the processing units whose addresses differ by 2 bidirectionally. Each of data bus group 660, 662, 664, 666, 668, 670, 672, 674 includes a pair of data buses in opposite directions.

[0163]

In the configuration shown in Figure 15, there are three inputs and three outputs as the data buses between one processing unit and another processing unit. This is the same for each processing unit. Also, the processing units and the data memories are classified into two groups, each of which includes four processing units and four data memories. The first group includes processing units 600,6 02, 604, 606 and data memories 620, 622, 624, 626. The second group includes processing unit 608, 610, 612, 614 and data memories 628, 630, 632, 634. The data memories and the processing units in each group are connected by memory read bus groups 682, 684, respectively. Also, each processing unit and memory write bus 680 are connected by a memory write bus group not shown in the figure so that data can be written from each processing unit to any of the data memories. Communication between the groups is carried out as described above via data memories.

[0164]

The data buses are grouped as shown in Figure 15 so that the read buses of the data memories can be shortened as much as possible. The speed will become slow as the read buses of the data memories are extended. When the processing units are grouped in a unit of 4 to shorten the length as described in this application example, the operation speed can be increased. In practical application, there are many operations that use processing units in a unit of 4, such as double precision operations and butterfly operations. Consequently, when the processing units and the data memories are grouped in a unit of 4 as shown in Figure 15, a processor that can handle a wide range of operations can be obtained.

[0165]

The processor shown in Figure 15 can perform 1 multiplication and 1 addition in double precision, butterfly, and sum-of-products operations of every 4 items in each group. Since the input data used in each processing unit are in that group, the data can be shared in the group. Consequently, it is also possible to include the data read buses from the data memories in the group as a variation from the configuration shown in Figure 15.

[0166]

In the processor shown in Figure 15, the processing units have exactly the same configuration. Consequently, the control instructions used for controlling each processing unit are interchangeable. Also, the layout of the processing units and the data memories can be simplified when manufacturing the processor. Also, when the selectors in each processing unit

are switched to realize the desired operation in the same way as described in the first application example, more complicated operations using plural processing units can be carried out than in the conventional technology.

[0167]

Of course, it is also possible to connect the data memory read bus group and the processing units in such a way that any processing unit can read data from any data memory. For example, a sum-of-products operation of every 8 items can be performed by using 8 processing units as one group instead of the grouping method shown in Figure 15. In this case, however, the speed of reading data from the data memories may decrease. Also, the grouping method shown in Figure 15 is preferred when performing an operation of every 4 items.

[0168]

Effects of the invention

Since the processing unit used for parallel processing described in Claim 1 can carry out various operations under the control of the control means, it is a general purpose processing unit, and the control of the control means is relatively simple. The number of the input/output data buses connected to the processing units is the same in all of the invention units, and each processing unit has the same configuration. Therefore, the processing units can be easily connected to each other when manufacturing a parallel processing processor by combining plural processing units. The control over these processing units is also interchangeable.

[0169]

As a result, it is possible to provide a processing unit for parallel processing that can carry out more kinds of operations under simple control.

[0170]

The processing unit described in Claim 2 uses a nxn-bit multiplier and 2 n+n-bit adders so that it is possible to carry out typical processing in an operation requiring a large amount data processing.

[0171]

As a result, it is possible to provide a processing unit for parallel processing that can carry out more kinds of operations under simple control.

[0172]

For the processing unit described in Claim 3, an operation between 2n-bit data can be carried out efficiently by dividing the data into high-order n bits and low-order n bits and by using an nxn-bit multiplier and n+n-bit adder .

[0173]

As a result, it is possible to provide a processing unit for parallel processing that can carry out more kinds of operations, including double precision operation, under simple control.

[0174]

For the processing unit described in Claim 4, it is possible to select the case when the carry output of one of the two adders is used or not used as the carry input of the other adder. Consequently, it is possible to perform addition of 2n-bit data and two different additions for n-bit data.

[0175]

As a result, it is possible to provide a processing unit for parallel processing that can carry out more kinds of large-quantity operations, including double precision operation, under simple control.

[0176]

The processing unit described in Claim 5 can be used to efficiently carry out both an operation using the addition result of its own adder, such as a sum-of-products operation, and an addition using the data provided via the data bus.

[0177]

As a result, it is possible to provide a processing unit for parallel processing that can carry out more kinds of large-quantity operations, including sum-of-products operation, under simple control.

[0178]

The processing unit described in Claim 6 can be used to efficiently carry out both an operation using the multiplication result to the input of addition, such as a sum-of-products operation, and an addition using the data provided via the data bus.

[0179]

As a result, it is possible to provide a processing unit for parallel processing that can carry out more kinds of large-quantity operations, including sum-of-products operation, under simple control.

[0180]

The processing unit described in Claim 7 can be used to efficiently carry out both an operation using the multiplication result or addition result as the input of addition, such as a sum-of-products operation, and an addition using the data provided via the data bus.

[0181]

As a result, it is possible to provide a processing unit for parallel processing that can carry out more kinds of large-quantity operations, including sum-of-products operation, under simple control.

[0182]

By using the multiplier in the processing unit described in Claim 8, it is possible to freely select the configuration used for efficiently performing high-speed operation by using the data stored in the read only memory means, such as division using the Newton-Raphson method and extraction-of-the-square-root operation, and the configuration used for performing the normal operation.

[0183]

As a result, it is possible to provide a processing unit for parallel processing that can carry out more kinds of large-quantity operations, including Newton-Raphon method and extraction-of-the-square-root operation, under simple control.

[0184]

For the parallel processing processor described in Claim 9, each processing unit has the same number of inputs and outputs and has at least two inputs and two outputs. The degree of freedom of the operation is increases as the number of inputs/outputs is increased. Also, since the processing units have the same configuration, the complexity in layout and control of the processor can be reduced.

[0185]

As a result, it is possible to provide a parallel processing processor with simple configuration, which can realize more kinds of operations under simple control.

[0186]

Each processing unit in the parallel processing processor described in Claim 10 carries out a prescribed processing using plural operation means to the data read from the data memory means or the data sent from another processing unit so that the output of any operation means an be output to another processing unit and any data storage means. Consequently, various operations can be carried out by plural processing units with the same configuration using interchangeable control instructions.

[0187]

As a result, it is possible to provide a parallel processing processor, which can realize more kinds of operations under simple control.

[0188]

In the parallel processing processor described in Claim 11, the read data bus from the data storage means to the processing unit is shorter than that in the case when all of the processing units are connected without being grouped. Therefore, data can be transferred at a high speed to the processing unit. Also, an operation using common data can be carried out using the processing units in one group. In addition, since data can be written from each processing unit to any data storage means, it is also possible to exchange data between the groups. Various operations can be carried out using plural processing units. Also, each processing unit has the same configuration and can be controlled using interchangeable control instructions.

[0189]

As a result, it is possible to provide a parallel processing processor, which can carry out more kinds of operations at high speed under simple control.

[0190]

In the operation method for parallel processing processor described in Claim 12, the high-order n bits or low-order n bits of the computation result can be provided to the multiplier or adders as low-order n bits or high-order n bits of data used for new operation. Various operations can be performed. Also, since it is possible to select whether to input/output carry between two adders, it is possible to switch between two addition processing with n-bit precision

and one addition processing with 2n-bit precision. Consequently, processing can be carried out with a very high degree of freedom. Also, each processing unit has the same configuration and can be controlled using interchangeable control instructions.

[0191]

As a result, it is possible to provide an operation method for parallel processing processor, which can carry out more kinds of operations under simple control.

[0192]

In the operation method for parallel processing processor described in Claim 13, the connection between the data buses and multiplier or adders can be set such that the high-order n bits or low-order n bits in the output of the multiplier in a certain processing unit can be input to either the low-order n bits or high-order n bits of the multiplier or adder in another processing unit. Various operations can be carried out without using a shift means. Also, the operation can be carried out at higher speed, and the control is more simple than the case using a shift means.

[0193]

As a result, it is possible to provide an operation method for parallel processing processor, which can carry out more kinds of operations at high speed under simple control.

[0194]

In the operation method for parallel processing processor described in Claim 14, the connection between the data buses and multiplier or adders can be set such that the high-order n bits in the output of the multiplier or adder in a certain processing unit can be input to either the low-order n bits or high-order n bits of the multiplier or adder in another processing unit. Data can be well shifted without using a shift means, and there is no need to spend time shifting. Also, the operation can be carried out at higher speed, and the control is more simple than the case using a shift means.

[0195]

As a result, it is possible to provide an operation method for parallel processing processor, which can carry out more kinds of operations at high speed under simple control.

[0196]

For the parallel processing processor described in Claim 15, a complicated operation can be controlled for each processing unit, and the overall control of the parallel processing

processor becomes easy. Also, in each processing unit, various kinds of operation processing are performed by using the control means. When plural processing units are combined, the entire processor can carry out more operations than the conventional processor.

[0197]

As a result, it is possible to provide a parallel processing processor that can realize more kinds of operations under simple control.

[0198]

For the parallel processing processor described in Claim 16, when a certain instruction is executed repeatedly while changing the data output from the data storage means, if the instruction used for controlling the data storage means is issued repeatedly while changing its content, there is no need to issue several different control instructions used for the operation means. Consequently, the control is simplified when performing the same operation with respect to a large amount of data. Also, various operations can be carried out in each processing unit by combining the control instructions. More complicated operation can be performed by combining plural processing units.

[0199]

As a result, it is possible to provide a parallel processing processor that can realize more kinds of operations under simple control.

[0200]

For the parallel processing processor described in Claim 17, a prescribed operation is carried out by reading two data from two read addresses of the data storage means. The result can be written at the position specified by one write address. Complicated operation can be carried out while data are transferred between plural processing units via data storage means. Each processing unit has the same configuration, and the layout is simple. Also, the processing units can be controlled under interchangeable instructions, and the control over the processor can be simplified.

[0201]

As a result, it is possible to provide a parallel processing processor that can realize more kinds of operations under simple control.

[0202]

For the parallel processing processor described in Claim 18, when a certain instruction is executed repeatedly while changing the data output from the data storage means, if the instruction used for controlling the data storage means is issued repeatedly while changing its content, there is no need to issue several different control instructions used for the operation means. Consequently, the control is simplified when performing the same operation with respect to a large amount of data. Also, various operations can be carried out in each processing unit by combining the control instructions. More complicated operation can be performed by combining plural processing units.

[0203]

As a result, it is possible to provide a parallel processing processor that can realize more kinds of operations under simple control.

[0204]

For the parallel processing processor described in Claim 19, if the two instructions read from the instruction memory are of the same system, they are executed at the same time. If they are of different systems, the processing is carried out in the normal way. The instruction execution speed can be increased compared with the case in which the control instruction is read out one at a time.

[0205]

As a result, it is possible to provide a parallel processing processor that can realize more kinds of operations at high speed under simple control.

Brief description of the figures

Figure 1 is a schematic block diagram illustrating the configuration of the processor disclosed in the first application example of the present invention.

Figure 2 is the block diagram of the processor disclosed in the first application example.

Figure 3 is the block diagram of the processing unit in the first application example.

Figure 4 is a block diagram illustrating connection between the data memory and the memory read data bus group in the first application example.

Figure 5 is a schematic diagram illustrating the structure of a control instruction.

Figure 6 is a schematic diagram illustrating the storage state of instructions in the instruction memory.

Figure 7 is a schematic block diagram illustrating the first connection example for the processor in the first application example of the present invention.

Figure 8 is a schematic block diagram illustrating the second connection example for the processor in the first application example of the present invention.

Figure 9 is a schematic block diagram illustrating the third connection example for the processor in the first application example of the present invention.

Figure 10 is a schematic block diagram illustrating the fourth connection example for the processor in the first application example of the present invention.

Figure 11 is a schematic block diagram illustrating the fifth connection example for the processor in the first application example of the present invention.

Figure 12 is the block diagram of the processor in the second application example of the present invention.

Figure 13 is a block diagram illustrating the main parts of the processor in the third application example of the present invention.

Figure 14 is the schematic block diagram of the instruction memory in the third application example of the present invention.

Figure 15 is a schematic block diagram illustrating the configuration of the processor in the fourth application example of the present invention.

Figure 16 is a schematic block diagram illustrating the conventional parallel processing processor.

Figure 17 is a block diagram illustrating the configuration of each processing unit in the conventional processor shown in Figure 16.

Figure 18 is a schematic block diagram illustrating a connection example of the processing units on the processing processor shown in Figures 16 and 17.

Figure 19 is a schematic block diagram illustrating a connection example between the processing units in the conventional processor.

Explanation of symbols

200, 202, 204, 206	Processing units
210, 212, 214, 216	Monodirectional input data bus
220, 22, 224, 226	Monodirectional input data bus
250, 252, 254, 256	Control circuit
260, 262, 264, 266	Instruction memory
280	Memory read data bus group
282	Memory write data bus group
270, 272, 274, 276	Data memory

350, 352, 354, 356	Selector
360, 362, 364, 366, 368, 370	Selector
380	Multiplier
383, 384	Adder
390	Selector
392	Crossbar switch

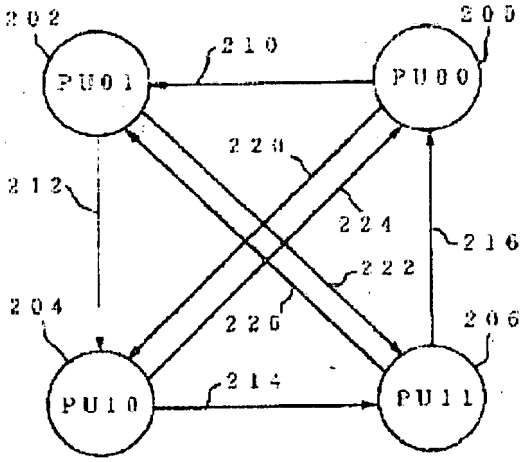


Figure 1

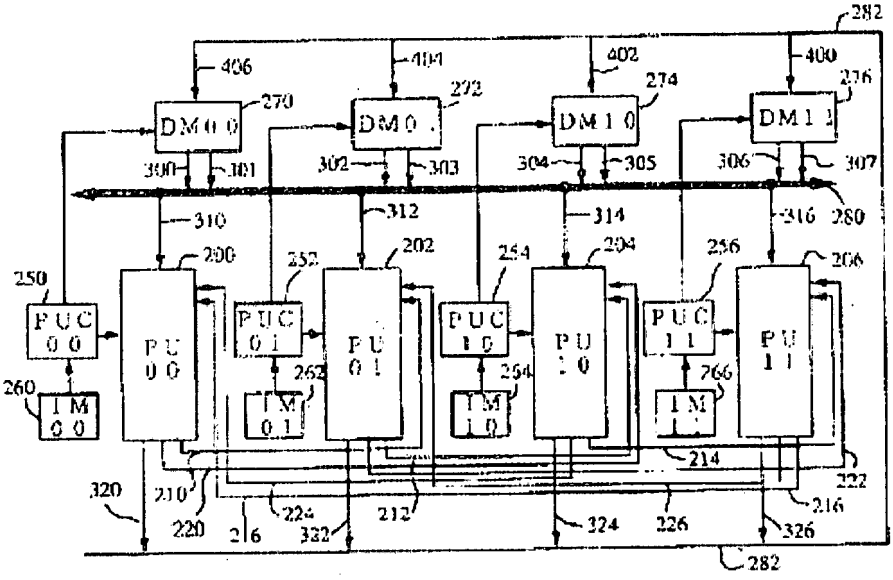


Figure 2

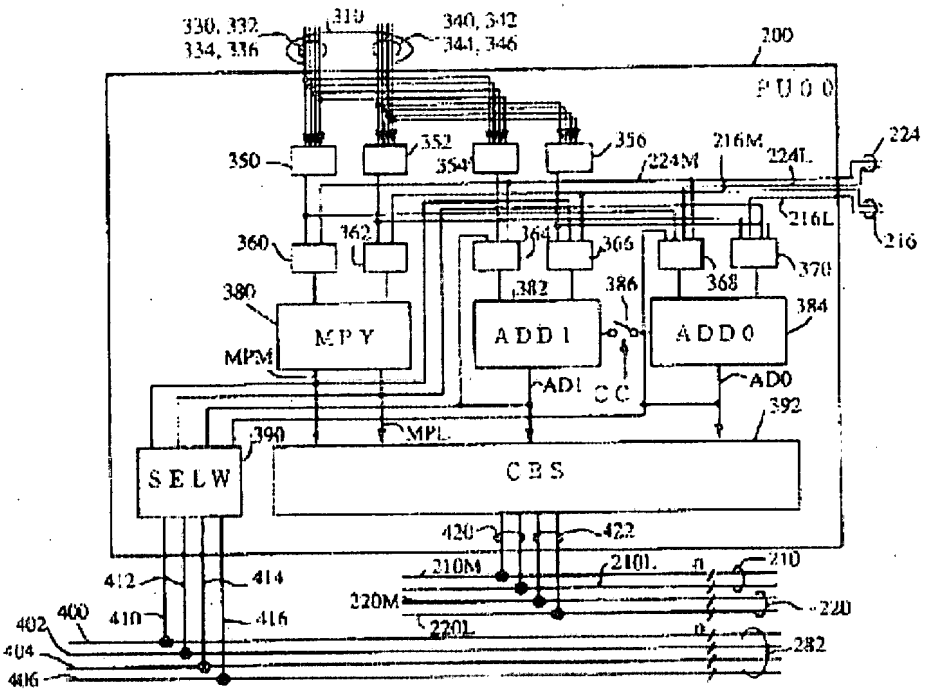


Figure 3

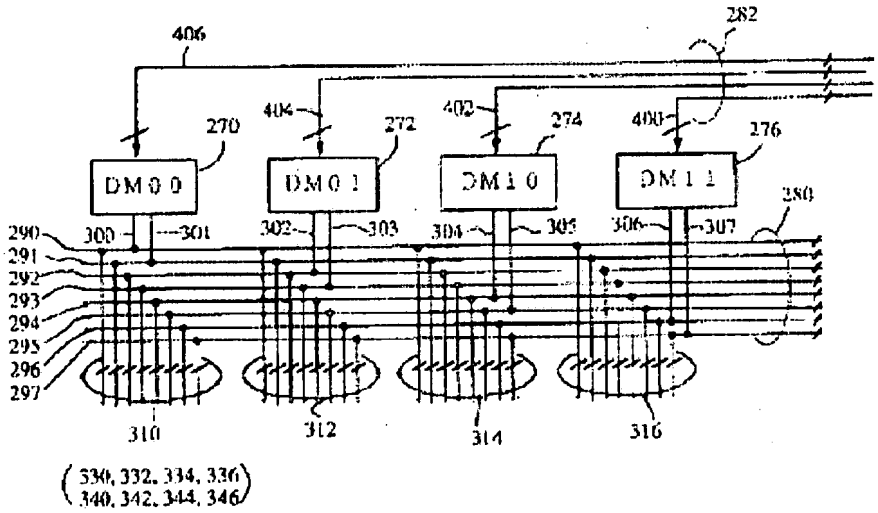


Figure 4

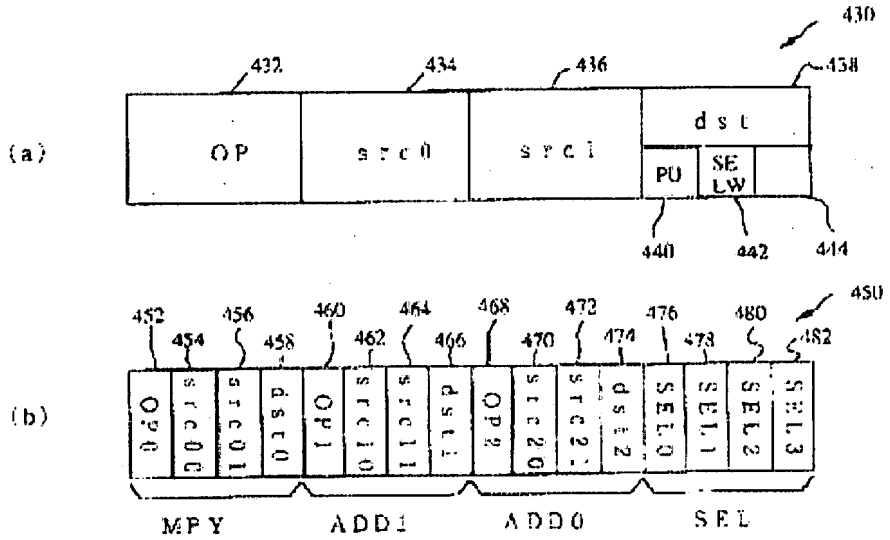


Figure 5

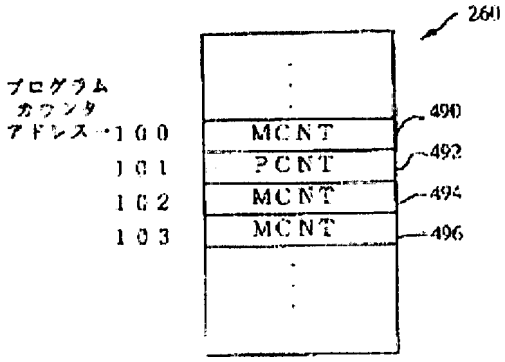


Figure 6

Key: 100 Program counter address

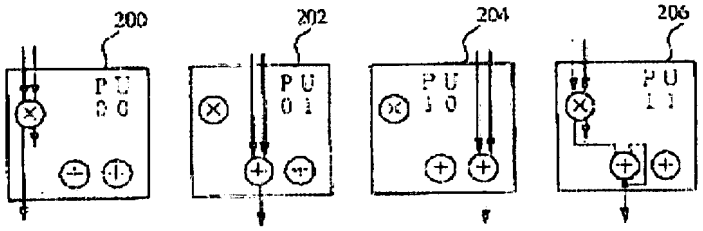


Figure 7

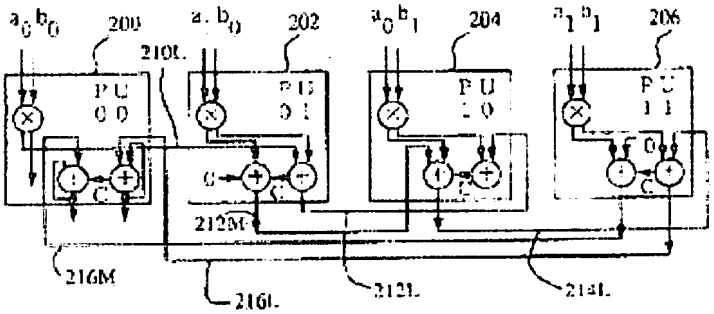


Figure 8

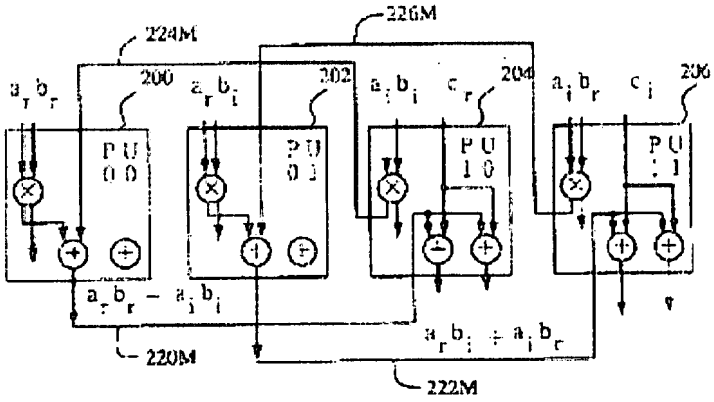


Figure 9

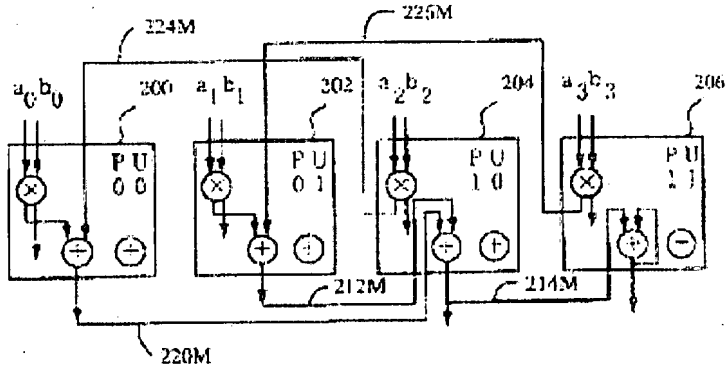


Figure 10

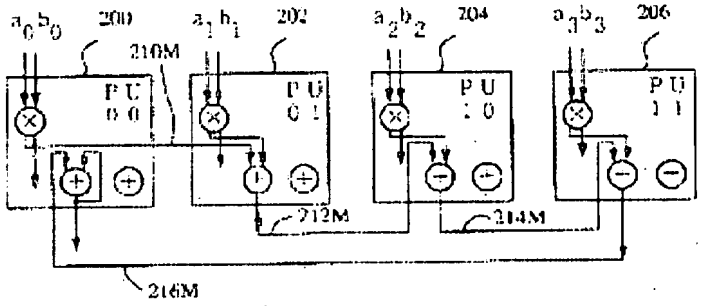


Figure 11

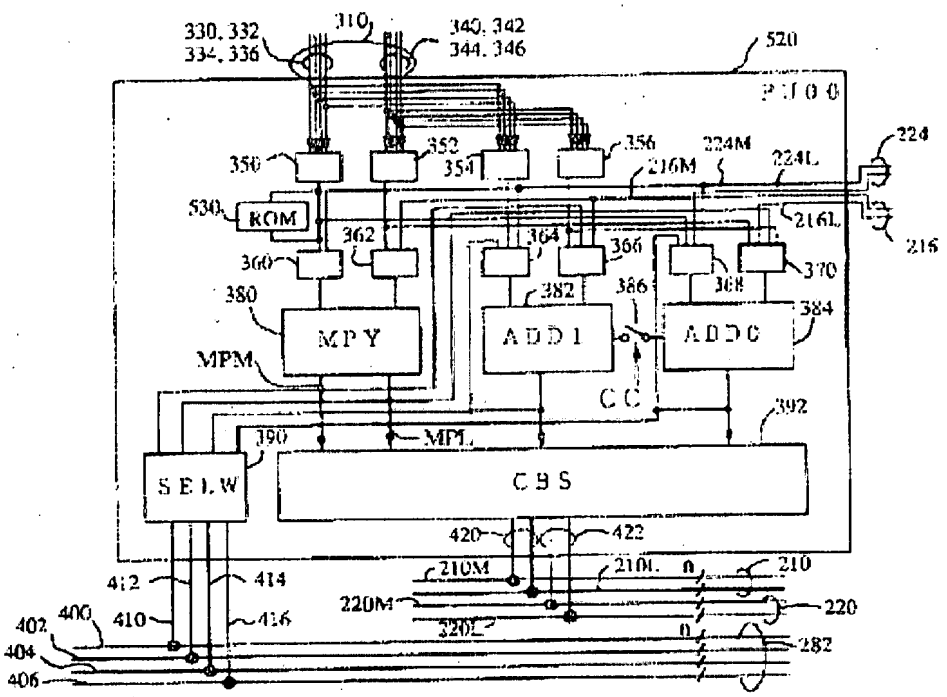


Figure 12

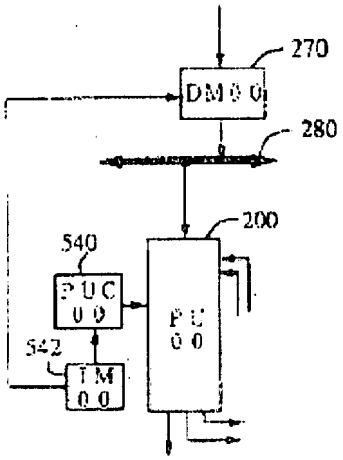


Figure 13

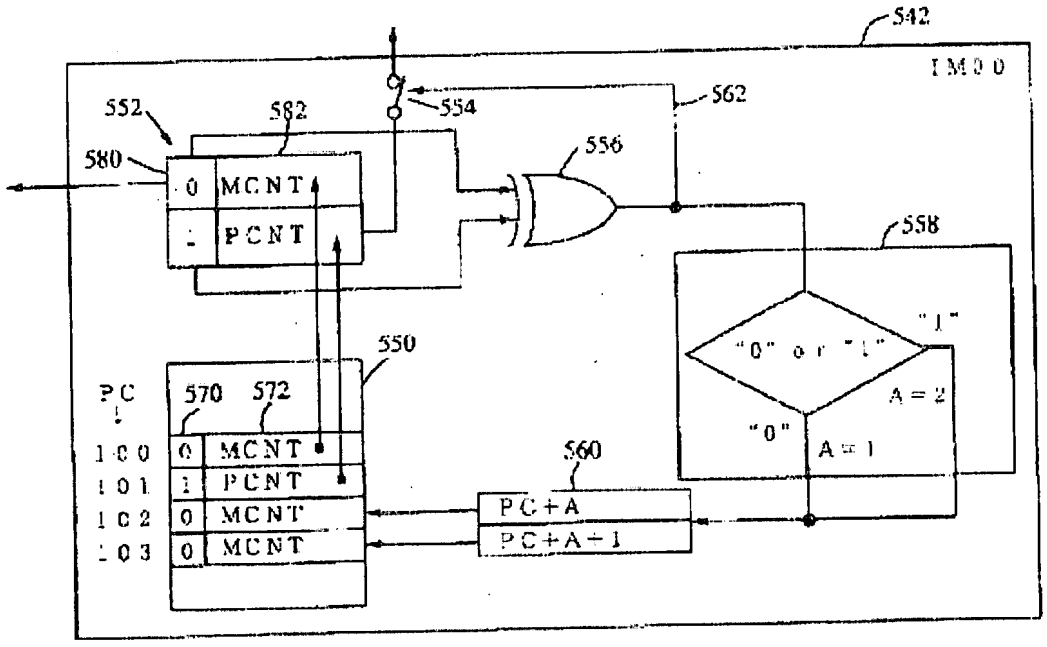


Figure 14

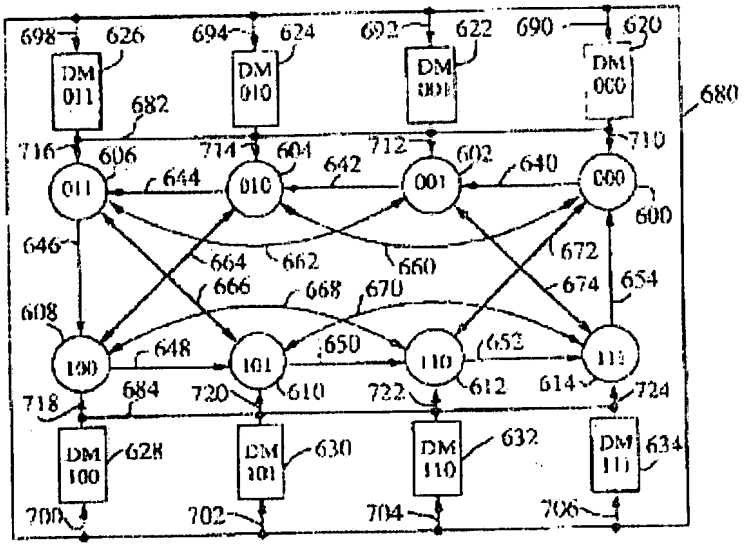


Figure 15

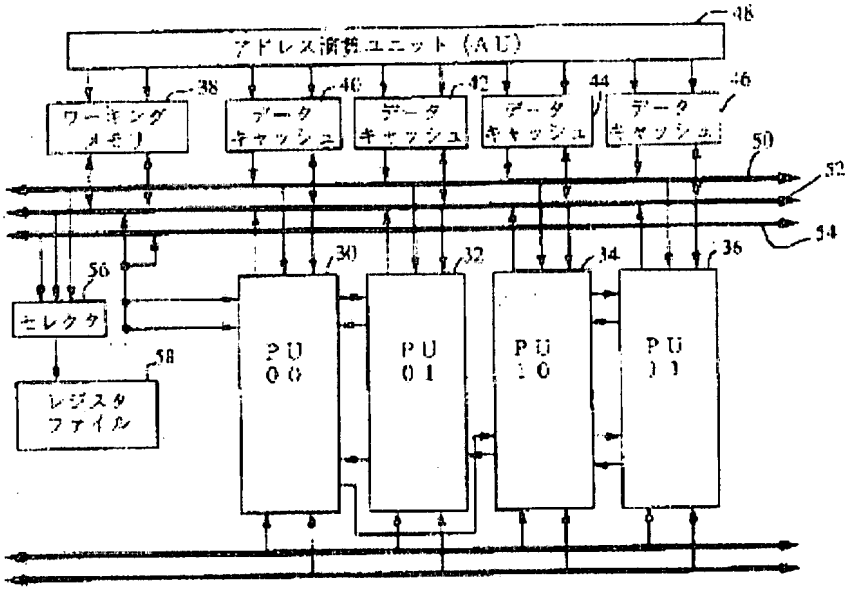


Figure 16

Key:
48	Address computing unit
38	Working memory
40, 42, 44, 46	Data cache
56	Selector
58	Register file

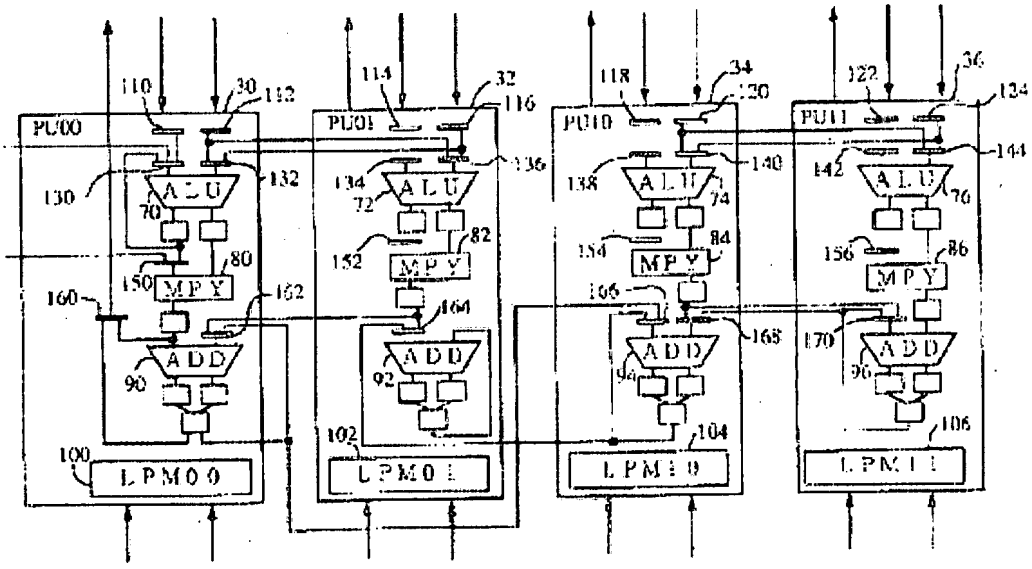


Figure 17

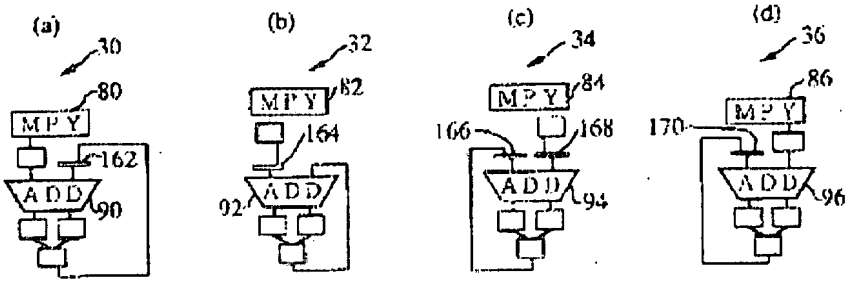


Figure 18

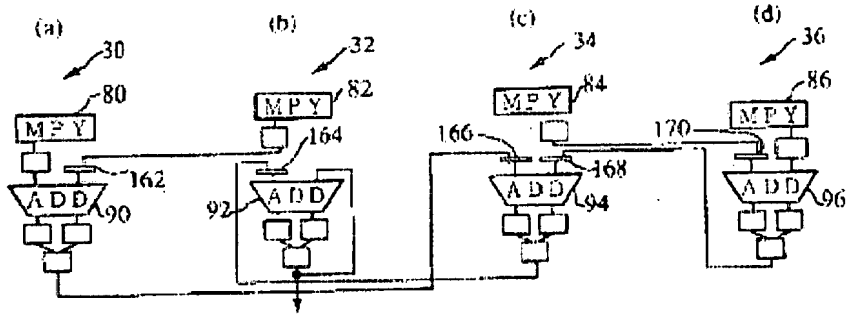


Figure 19